

Attorney Docket No. 1075.11

#2  
a/f  
12/12/01

U.S. PTO  
09/986818  
11/13/01

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Patent Application of:

Miki TAKAGI et al.

Application No.: Unassigned

Group Art Unit: Unassigned

Filed: November 12, 2001

Examiner:

For: AUTOMATED HDL MODIFYING APPARATUS AND COMPUTER-READABLE  
RECORDING MEDIUM IN WHICH PROGRAM FOR AUTOMATICALLY MODIFYING  
HDL IS RECORDED

**SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN  
APPLICATION IN ACCORDANCE  
WITH THE REQUIREMENTS OF 37 C.F.R. § 1.55**

Assistant Commissioner for Patents  
Washington, D.C. 20231

Sir:

In accordance with the provisions of 37 C.F.R. § 1.55, the applicant(s) submit(s)  
herewith a certified copy of the following foreign application:

Japanese Patent Application No. 2001-189337

Filed: June 22, 2001

It is respectfully requested that the applicant(s) be given the benefit of the foreign filing  
date(s) as evidenced by the certified papers attached hereto, in accordance with the  
requirements of 35 U.S.C. § 119.

Respectfully submitted,  
STAAS & HALSEY LLP

Date: November 12, 2001

By: \_\_\_\_\_

James D. Halsey, Jr.  
Registration No. 22,729

700 11th Street, N.W., Ste. 500  
Washington, D.C. 20001  
(202) 434-1500

日 本 国 特 許 庁  
JAPAN PATENT OFFICE



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2001年 6月22日

出 願 番 号

Application Number:

特願2001-189337

出 願 人

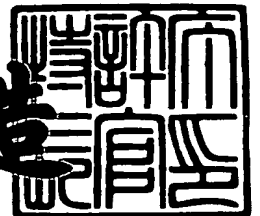
Applicant(s):

富士通株式会社

2001年 8月10日

特許庁長官  
Commissioner,  
Japan Patent Office

及 川 耕 造



出証番号 出証特2001-3072257

【書類名】 特許願

【整理番号】 0150589

【提出日】 平成13年 6月22日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 17/50

【発明の名称】 H D L 自動修正装置および H D L 自動修正プログラム並びに同プログラムを記録したコンピュータ読取可能な記録媒体

【請求項の数】 5

【発明者】

    【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

    【氏名】 ▲高▼木 美紀

【発明者】

    【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

    【氏名】 竹山 広治

【発明者】

    【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

    【氏名】 野口 弘

【特許出願人】

    【識別番号】 000005223

    【氏名又は名称】 富士通株式会社

【代理人】

    【識別番号】 100092978

    【弁理士】

    【氏名又は名称】 真田 有

    【電話番号】 0422-21-4222

【手数料の表示】

【予納台帳番号】 007696

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9704824

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 HDL自動修正装置およびHDL自動修正プログラム並びに  
同プログラムを記録したコンピュータ読取可能な記録媒体

【特許請求の範囲】

【請求項1】 ハードウェア記述言語（以下、HDLという）によって記述された回路設計情報（以下、HDL記述という）を自動的に修正する装置であって、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段と、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段と、

該HDL構文解析手段による解析結果に基づいて該HDL記述の意味論解析を行ない、該HDL記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段と、

前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を、型変換規則として定義する型変換テンプレートと、

該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代入文の右辺に対して、該型変換テンプレートに定義された該型変換関数を適用することにより、該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段と、

該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段と、

該意味論的文法違反修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段とをそなえて構成されたことを特徴とする、HDL自動修正装置。

【請求項2】 文法違反ではないが回路設計上考慮すべき修正対象事項、および、該修正対象事項に対応する箇所の修正規則を定義する制御情報テンプレートと、

該HDL構文解析手段による解析結果に基づき、該HDL記述において、該修

正対象事項に対応する箇所を検出する修正対象事項検出手段と、

該修正対象事項検出手段によって検出された箇所を、該制御情報テンプレートに定義された該修正規則に従って修正する修正対象事項修正手段とをさらにそなえ、

該HDL逆構文解析手段が、該意味論的文法違反修正手段および該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行なうとともに、

該コメント付与手段が、該意味論的文法違反修正手段および該修正対象事項修正手段によって修正を施された箇所に、該コメントを付与することを特徴とする、請求項1記載のHDL自動修正装置。

【請求項3】 ハードウェア記述言語（以下、HDLという）によって記述された回路設計情報（以下、HDL記述という）を自動的に修正する装置であって、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段と、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段と、

文法違反ではないが回路設計上考慮すべき修正対象事項、および、該修正対象事項に対応する箇所の修正規則を定義する制御情報テンプレートと、

該HDL構文解析手段による解析結果に基づき、該HDL記述において、該修正対象事項に対応する箇所を検出する修正対象事項検出手段と、

該修正対象事項検出手段によって検出された箇所を、該制御情報テンプレートに定義された該修正規則に従って修正する修正対象事項修正手段と、

該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段と、

該修正対象事項修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段とをそなえて構成されたことを特徴とする、HDL自動修正装置。

【請求項4】 ハードウェア記述言語（以下、HDLという）によって記述された回路設計情報（以下、HDL記述という）を、コンピュータに自動修正さ

せるためのHDL自動修正プログラムであって、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段、

該HDL構文解析手段による解析結果に基づいて該HDL記述の意味論解析を行ない、該HDL記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段、

該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代入文の右辺に対して、前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を適用することにより、該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段、

該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段、および、

該意味論的文法違反修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段として、該コンピュータを機能させることを特徴とする、HDL自動修正プログラム。

【請求項5】 ハードウェア記述言語（以下、HDLという）によって記述された回路設計情報（以下、HDL記述という）を、コンピュータに自動修正させるためのHDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体であって、

該HDL自動修正プログラムが、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段、

該HDL構文解析手段による解析結果に基づいて該HDL記述の意味論解析を行ない、該HDL記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段、

該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代

入文の右辺に対して、前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を適用することにより、該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段、

該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段、および、

該意味論的文法違反修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段として、該コンピュータを機能させることを特徴とする、HDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、例えばVHDL〔VHSIC (Very High Speed Integrated Circuit) HDL〕やVerilog-HDLなどのハードウェア記述言語〔以下、HDL (Hardware Discription Language) という〕によって記述された、電子システムや論理回路の設計情報における、不適切な記述を自動修正するための装置およびプログラム並びに同プログラムを記録したコンピュータ読取可能な記録媒体に関する。

【0002】

【従来の技術】

LSI (Large Scale Integration) 等の半導体集積回路を設計する際、その半導体集積回路の論理回路についての設計情報を、VHDLやVerilog-HDL等のHDLによって記述することが行なわれている。HDLは、一般的なソフトウェアプログラミング言語とは異なるもので、電子システムや論理回路の機能や構造を記述するのに適し、設計レベルを論理ゲートレベルからマイクロアーキテクチャレベルに高め階層設計を行なうのに適している。そして、上述のごとくHDLによって記述された設計情報（以下、HDL記述という場合がある）に基づいて、論理回路（ネットリスト）が論理合成ツールにより自動的に合成される。



## 【 0 0 0 3 】

このような論理合成を行なう前、つまり、HDL記述を論理合成ツールに入力する前には、HDL記述における文法違反のチェック（構文チェック）が行なわれている。この文法違反のチェックに際しては、例えば下記公報に開示された技術を用いることができる。

## 【 0 0 0 4 】

例えば特開平 2 - 2 9 4 7 3 6 号公報では、ソフトウェアプログラミング言語で記述されたソースプログラムにおいて、例えば文末に記入されるべきセミコロンの “ ; ” が抜けている等の、比較的軽度の構文的文法違反を検出し、その文法違反を自動修正する技術が開示されている。この技術を用いることにより、ソースプログラム中の文末に “ ; ” が抜けている場合、その “ ; ” が自動的に挿入される。

## 【 0 0 0 5 】

また、例えば特開平 6 - 4 4 0 8 1 号公報では、PASCAL言語で記述された入力ソースプログラムについて、代入文の右辺と左辺における変数の型が不一致である場合に、その旨を通知するエラーメッセージを出力する技術が開示されている。

## 【 0 0 0 6 】

## 【発明が解決しようとする課題】

特開平 2 - 2 9 4 7 3 6 号公報に開示された技術を用いてHDL記述の文法違反チェックを行なった場合、上述のような構文的文法違反（文法違反としては軽度なもの）については自動修正することはできるが、代入文の右辺と左辺における変数の型が一致しないといった、意味論的文法違反（文法違反としては重度なもの）を自動修正することができない。

## 【 0 0 0 7 】

従って、意味論的文法違反については、設計者（ユーザ）が、HDL処理系から出力されるエラーメッセージを参照して手修正しなければならない。その際、エラーメッセージは、具体的にどのような修正を行なえば良いかを示すものではないため、1回の修正で意味論的文法違反を解消できない場合が多く、設計者に

かかる負担が大きいという課題があった。特に、上述のような意味論的文法違反は、比較的高い頻度で発生するため、その文法違反を修正するために設計者にかかる負担はかなり大きいものとなる。

## 【 0 0 0 8 】

また、HDL記述が自動修正された場合、設計者は、自動修正結果が意図するものかどうか、即ち適正な修正が行なわれたか否かを確認する必要があるが、従来技術では、修正箇所を示す目印等が自動修正結果に含まれていないので、HDL記述のどこをどのように修正したかを確認するのに手間がかかり、設計者にかかる負担が大きいという課題もあった。

## 【 0 0 0 9 】

特開平6-44081号公報に開示された技術を用いてHDL記述の文法違反チェックを行なった場合も、設計者は、代入文の右辺と左辺における変数の型が不一致である意味論的文法違反の発生を、エラーメッセージによって認識することはできるが、その文法違反を自動的に修正することができない。また、設計者は、エラーメッセージにより文法違反箇所および違反内容を知ることができても、その文法違反にどのように対処したらよいか分からず、文法書等を参照して文法違反の解消手法を検討しながら手修正する必要があるため、設計者にかかる負担は極めて大きい。

## 【 0 0 1 0 】

一方、従来技術では、文法違反をチェックすることができても、文法違反ではないが回路設計を行なう上で考慮すべき箇所（不適切な記述）をチェックして自動的に適切な記述に修正することができなかった。

従来、文法違反ではないが回路設計を行なう上で考慮すべき、名前の命名規則や論理合成記述規則（論理合成が可能な記述に関する規則）などに違反する箇所を検出しその違反内容を出力するスタイルチェッカも提案されているが、このスタイルチェッカは、違反内容を出力するだけで、その違反を回避するようにHDL記述を自動修正することはできない。従って、設計者は、スタイルチェッカからのエラーメッセージを参照して手修正でその違反を解消しなければならず、設計者にかかる負担が大きい。特に、複数の設計者によりHDLを用いて回路設計

を行なった場合などには、命名規則に違反する箇所が多数出現する可能性が高く、違反箇所の名前を修正するのに多大な労力を要することになる。

#### 【0011】

また、文法違反ではないが回路の階層設計上考慮すべき規則や配線の接続関係の規則に違反するHDL記述のいくつかは、従来、フロントエンド（言語処理系）ではなく、バックエンド（論理合成ツールや検証ツール）でないと検出できない。そのようなHDL記述は、設計者の単純ミスによるものであるにもかかわらず早期に発見できないため、当然、自動修正することもできず、設計工程の手戻りの発生要因となっている。例えば、HDL記述が複数の階層から構成されている場合、各階層での端子定義記述とインスタンス部における端子記述とが不一致になることが多々ある。このような不一致は、ミスによって生じる場合もあるが、文法上あるいは回路表現上において全く問題ない場合もある。しかし、上記不一致は、後で回路設計上の問題を生じさせ上述のような手戻りの発生要因となるおそれがあるため、上記不一致を含まない記述にHDL記述を自動修正できるようにすることが望まれている。

#### 【0012】

さらに、HDLを用いて回路設計を行なった場合、システム間の関係や設計フローの都合などにより、当初用いられていたHDLから異なる種類の他のHDLにHDL記述を変換することがしばしばある。このとき、当然、HDL記述が現在のHDLの言語規約を満たしていても他のHDLの言語規約を満たさない場合があるため、上述のような変換の可能性がある場合、他のHDLに変換した時に回路設計上の問題が生じないように、予め他のHDLの言語規約も満たすようにHDL記述を自動修正できるようにすることが望まれている。

#### 【0013】

また、論理合成したいHDL記述においては、論理検証時に使用した論理合成不能な波形観測用シミュレーション記述等がコメントアウトされずに残っている場合が多々ある。このような論理合成不能な記述についても、自動修正（自動削除）する手立てがなく、従来、設計者が、HDL記述を参照し論理合成不能な記述を見つけた場合に手修正するか、論理合成ツールでその記述によってエラーが

発生してから何らかの対処をとるしかなく、いずれにしても設計者にとって大きな負担となっていた。

#### 【 0 0 1 4 】

本発明は、このような課題に鑑み創案されたもので、重度の意味論的文法違反を自動修正し且つその修正箇所が明確になるようにするほか、文法違反ではないが回路設計上考慮すべき箇所を適切な記述に自動修正し且つその修正箇所が明確になるようにして、設計者にかかる負担を大幅に軽減するとともに、高品質のHDL記述が得られるようにした、HDL自動修正装置およびHDL自動修正プログラム並びに同プログラムを記録したコンピュータ読取可能な記録媒体を提供することを目的とする。

#### 【 0 0 1 5 】

##### 【課題を解決するための手段】

上記目的を達成するために、本発明のHDL自動修正装置（請求項1）は、HDLで記述されたHDL記述（回路設計情報）を自動的に修正する装置であって、修正対象のHDL記述の字句解析を行なうHDL字句解析手段と、該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない該HDL記述を解析木形式の記述に変換するHDL構文解析手段と、該HDL構文解析手段による解析結果に基づいて該HDL記述の意味論解析を行ない該HDL記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段と、前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を型変換規則として定義する型変換テンプレートと、該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代入文の右辺に対して、該型変換テンプレートに定義された該型変換関数を適用することにより該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段と、該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段と、該意味論的文法違反修正手段によって修正を施された箇所に該修正に関するコメントを付与するコメント付与手段とをそなえて構成されたことを特徴としている。

## 【 0 0 1 6 】

このとき、文法違反ではないが回路設計上考慮すべき修正対象事項および該修正対象事項に対応する箇所の修正規則を定義する制御情報テンプレートと、該HDL構文解析手段による解析結果に基づき、該HDL記述において該修正対象事項に対応する箇所を検出する修正対象事項検出手段と、該修正対象事項検出手段によって検出された箇所を該制御情報テンプレートに定義された該修正規則に従って修正する修正対象事項修正手段とをさらにそなえ、該HDL逆構文解析手段が、該意味論的文法違反修正手段および該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行なうとともに、該コメント付与手段が、該意味論的文法違反修正手段および該修正対象事項修正手段によって修正を施された箇所に、該コメントを付与するように構成してもよい（請求項2）。

## 【 0 0 1 7 】

また、本発明のHDL自動修正装置（請求項3）は、HDLによって記述されたHDL記述を自動的に修正する装置であって、上述したHDL字句解析手段およびHDL構文解析手段をそなえとともに、文法違反ではないが回路設計上考慮すべき修正対象事項および該修正対象事項に対応する箇所の修正規則を定義する制御情報テンプレートと、該HDL構文解析手段による解析結果に基づき該HDL記述において該修正対象事項に対応する箇所を検出する修正対象事項検出手段と、該修正対象事項検出手段によって検出された箇所を、該制御情報テンプレートに定義された該修正規則に従って修正する修正対象事項修正手段と、該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行ない該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段と、該修正対象事項修正手段によって修正を施された箇所に該修正に関するコメントを付与するコメント付与手段とをそなえて構成されたことを特徴としている。

## 【 0 0 1 8 】

さらに、本発明のHDL自動修正プログラム（請求項4）は、HDLによって記述されたHDL記述を、コンピュータに自動修正させるためのものであって、上述したHDL字句解析手段、HDL構文解析手段、意味論的文法違反検出手段

、意味論的文法違反修正手段、HDL逆構文解析手段およびコメント付与手段として、該コンピュータを機能させることを特徴としている。そして、本発明のコンピュータ読取可能な記録媒体（請求項5）は、上述したHDL自動修正プログラムを記録したものである。

【0019】

【発明の実施の形態】

以下、図面を参照して本発明の実施の形態を説明する。

〔1〕本発明の一実施形態の構成の説明

図1は本発明の一実施形態としてのHDL自動修正装置の構成を示すブロック図であり、この図1に示すように、本実施形態のHDL自動修正装置1は、HDLによって記述されたHDL記述（回路設計情報）を自動的に修正するものであって、HDL字句解析手段11、HDL構文解析手段12、構文的文法違反検出手段13、構文的文法違反修正手段14、意味論的文法違反検出手段15、意味論的文法違反修正手段16、修正対象事項検出手段17、修正対象事項修正手段18、HDL逆構文解析手段19、コメント付与手段20、データベース21～24およびテンプレート30、40、51～55をそなえて構成されている。

【0020】

HDL字句解析手段11は、修正対象のHDL記述（初期HDL記述）2Aの字句解析を行なうもので、HDL記述2Aを文字列の基本単位（トークン）に分解し、その基本単位を各基本単位の種別に関する情報とともにトークンデータベース21に書き込むものである。

HDL構文解析手段12は、トークンデータベース21のトークン（HDL字句解析手段11による解析結果）に基づいて、HDL記述2Aの構文解析を行ない、HDL記述2Aを解析木形式の記述に変換するものである。

【0021】

文法解析テンプレート30は、予約語の使用規則や、スペルについての規則や、構文規則などを定義するものである。ここで、構文規則は、例えば文中に“＝”があった場合、その文末には“；”が記入されるといった構文的文法についての規則である。

構文的文法違反検出手段 1 3 は、HDL 構文解析手段 1 2 による解析結果と文法解析テンプレート 3 0 とに基づいて、HDL 記述 2 A における構文的文法違反箇所（スペルの間違っている箇所や、“=”を記述する文の文末に“;”が抜けている箇所や、間違って予約語を使用している箇所など）を検出するもので、構文的文法違反修正手段 1 4 は、構文的文法違反検出手段 1 3 によって検出された構文的文法違反箇所を、文法解析テンプレート 3 0 によって定義された規則に従って、正しい記述に修正するものである。

## 【 0 0 2 2 】

この構文的文法違反修正手段 1 4 による修正結果が、HDL データベース 2 2 に書き込まれるようになっている。なお、構文的文法違反検出手段 1 3 によって構文的文法違反箇所が検出されなかった場合には、構文的文法違反修正手段 1 4 による修正処理は実行されず、HDL 構文解析手段 1 2 による解析結果がそのまま HDL データベース 2 2 に書き込まれることになる。

## 【 0 0 2 3 】

意味論的文法違反検出手段 1 5 は、HDL データベース 2 2 のデータ（HDL 構文解析手段 1 2 による解析結果もしくは構文的文法違反修正手段 1 4 による修正結果）に基づいて HDL 記述 2 A の意味論解析を行ない、HDL 記述 2 A における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出するものである。

## 【 0 0 2 4 】

型変換テンプレート 4 0 は、各種代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を、型変換規則として定義するもので、その型変換関数については、表 1 を参照しながら後述する。

意味論的文法違反修正手段 1 6 は、意味論的文法違反検出手段 1 5 によって意味論的文法違反箇所とみなされた代入文の右辺に対して、型変換テンプレート 4 0 に定義された型変換関数を適用することにより、意味論的文法違反箇所を正しい記述に修正するもので、その修正結果は HDL データベース 2 3 に書き込まれるようになっている。本実施形態の HDL 自動修正装置 1 による意味論的文法違反箇所の具体的な修正動作については、図 2 を参照しながら後述する。

## 【0025】

なお、意味論的文法違反検出手段15によって意味論的文法違反箇所が検出されなかった場合には、意味論的文法違反修正手段16による修正処理は実行されず、HDLデータベース22をそのままHDLデータベース23として用いることになる。

## 【0026】

テンプレート（制御情報テンプレート）51～55は、それぞれ、文法違反ではないが回路設計上考慮すべき修正対象事項、および、その修正対象事項に対応する箇所の修正規則を定義するもので、その詳細については後述する。

修正対象事項検出手段17は、HDLデータベース23のデータ（HDL構文解析手段12による解析結果もしくは構文的文法違反修正手段14／意味論的文法違反修正手段16による修正結果）に基づき、HDL記述2Aの中で、テンプレート51～55のそれぞれで定義された修正対象事項に対応する箇所を検出するものである。

## 【0027】

修正対象事項修正手段18は、修正対象事項検出手段17によって検出された箇所を、テンプレート51～55のそれぞれに定義された修正規則に従って修正するもので、その修正結果はHDLデータベース24に書き込まれるようになっている。

なお、修正対象事項検出手段17によって修正対象事項に対応する箇所が検出されなかった場合、修正対象事項修正手段18による修正処理は実行されず、HDLデータベース23をそのままHDLデータベース24として用いることになる。

## 【0028】

HDL逆構文解析手段19は、HDLデータベース24のデータ、つまり、構文的文法違反修正手段14、意味論的文法違反修正手段16や修正対象事項修正手段18による修正を施されたHDL記述（解析木形式の記述）についての逆構文解析を行ない、そのHDL記述を解析木形式の記述から通常の記述形式に変換し、その変換結果を修正後のHDL記述2Bとして出力するものである。



## 【 0 0 2 9 】

コメント付与手段 2 0 は、修正箇所（構文的文法違反修正手段 1 4，意味論的文法違反修正手段 1 6 や修正対象事項修正手段 1 8 によって修正を施された箇所）に、その修正に関するコメントを付与するものである。このコメント付与手段 2 0 によって H D L 記述 2 B に付与される具体的なコメント等については、図 2 ～図 9 を参照しながら後述する。なお、本実施形態の H D L 自動修正装置 1 によって初期 H D L 記述 2 A に対する修正が何ら施されなかった場合、コメント付与手段 2 0 は、当然、H D L 記述 2 B に対し何らコメントを付与しない。また、コメント付与手段 2 0 は、H D L 逆構文解析手段 1 9 から出力された H D L 記述 2 B（ソース）にコメントを付与する。

## 【 0 0 3 0 】

ここで、上述したテンプレート 5 1 ～5 5 について、より詳細に説明する。

言語変換規則テンプレート 5 1 における修正対象事項は、修正対象事項検出手段 1 7 が、処理中の H D L（例えば V H D L）を異なる種類の他の H D L（例えば Verilog-H D L）に変換した場合に他の H D L の言語規約を満たさない箇所を、修正対象事項に対応する箇所として検出するように定義されている。また、制御情報テンプレート 5 1 における修正規則は、修正対象事項修正手段 1 8 が、修正対象事項検出手段 1 7 によって検出された箇所を、他の H D L の言語規約を満たす記述に変換・修正するように定義されている。

## 【 0 0 3 1 】

ここで、言語変換規則テンプレート 5 1 は、図 3 に示すような予約語テンプレート 5 1 a，名前テンプレート 5 1 b，名前生成ルール 5 1 c および大文字小文字ルール 5 1 d を有している。予約語テンプレート 5 1 a には、各種 H D L ごとに使用可能な予約語が登録・定義され、名前テンプレート 5 1 b には、各種 H D L ごとに使用可能な端子名やネット名が登録・定義されている。また、名前生成ルール 5 1 c には、端子名やネット名を修正する際に、H D L 記述に存在する文字列と重複しないユニークな文字列を新たな端子名やネット名として生成するための規則が定義されている。さらに、大文字小文字ルール 5 1 d には、各種 H D L が、それぞれ、アルファベットの大文字と小文字を区別して使用するものであ

るか、もしくは、アルファベットの大文字と小文字を区別しないで使用するものであるかが登録・定義されている。

【 0 0 3 2 】

図 3 を参照しながら後述するごとく、例えば、予約語テンプレート 5 1 a および名前テンプレート 5 1 b を用い、例えば、Verilog-HDL による HDL 記述中に、Verilog-HDL では端子名やネット名として使用可能であるが VHDL では予約語として使用される文字列が検出された場合、その文字列が、VHDL による記述を行なった場合に予約語に該当しないような、新たな文字列に変換・修正される。その際、名前生成ルール 5 1 c を用いて、HDL 記述に存在する文字列と重複しないユニークな文字列が、新たな文字列として生成されるようになっている。

【 0 0 3 3 】

また、上述した大文字小文字ルール 5 1 d (図 3 参照) によって処理中の HDL がアルファベットの大文字と小文字を区別して使用するものであると認識された場合 (例えば Verilog-HDL)、処理中の HDL を、大文字と小文字を区別しないで使用する他の HDL に変換する可能性を考慮し、修正対象事項検出手段 1 7 が、大文字と小文字を区別して記述された同一綴りの一对の文字列のうちの一方を、修正対象事項に対応する箇所として検出するように、言語変換規則テンプレート 5 1 における修正対象事項を定義してもよい。その際、言語変換規則テンプレート 5 1 における修正規則は、修正対象事項修正手段 1 8 が、上述の名前生成ルール 5 1 c に従って、HDL 記述中に存在する文字列の綴りと重複しない綴りの新しい文字列を生成してから、修正対象事項検出手段 1 7 によって検出された一方の文字列を新しい文字列に置換するように定義される。

【 0 0 3 4 】

逆に、上述した大文字小文字ルール 5 1 d (図 3 参照) によって処理中の HDL がアルファベットの大文字と小文字を区別しないで使用するものであると認識された場合 (例えば VHDL)、処理中の HDL を、大文字と小文字を区別して使用する他の HDL に変換する可能性を考慮し、修正対象事項検出手段 1 7 が、HDL 記述の文字列における大文字および小文字のうちの一方を、修正対象事項

に対応する箇所として検出するように、言語変換規則テンプレート 5 1 における修正対象事項を定義してもよい。その際、修正対象事項修正手段 1 8 が、修正対象事項検出手段 1 7 によって検出された大文字または小文字をそれぞれ小文字または大文字に変換するように、言語変換規則テンプレート 5 1 における修正規則が定義される。

なお、上述した言語変換規則テンプレート 5 1 を用いた具体的な修正動作については、図 3 を参照しながら後述する。

#### 【 0 0 3 5 】

使用禁止文字情報テンプレート 5 2 における修正対象事項は、修正対象事項検出手段 1 7 が、所定の使用禁止文字を含む文字列を、修正対象事項に対応する箇所として検出するように定義されている。また、使用禁止文字情報テンプレート 5 2 における修正規則は、修正対象事項修正手段 1 8 が、HDL 記述に存在する文字列と重複せず且つ所定の使用禁止文字を含まない新しい文字列を生成してから、修正対象事項検出手段 1 7 によって検出された使用禁止文字を含む文字列を新しい文字列に置換するように定義されている。

#### 【 0 0 3 6 】

ここで、使用禁止文字情報テンプレート 5 2 は、図 4 に示すような使用禁止文字テンプレート 5 2 a および名前生成ルール 5 2 b を有している。使用禁止文字テンプレート 5 2 a には、使用を禁止されている文字列が定義・登録されており、修正対象事項検出手段 1 7 は、使用禁止文字テンプレート 5 2 a を参照して使用禁止文字を含む文字列を、修正対象事項に対応する箇所として検出するようになっている。また、名前生成ルール 5 2 b は、前述した名前生成ルール 5 1 c と同様のもので、HDL 記述に存在する文字列と重複せず且つ所定の使用禁止文字を含まないユニークな文字列を新たに生成するための規則が定義されており、修正対象事項修正手段 1 8 は、名前生成ルール 5 2 b に従って、上述した新しい文字列を生成している。

なお、上述した使用禁止文字情報テンプレート 5 2 を用いた具体的な修正動作については、図 4 を参照しながら後述する。

#### 【 0 0 3 7 】

階層情報テンプレート 5 3 における修正対象事項は、修正対象事項検出手段 1 7 が、HDL 記述を成す複数の階層において端子記述が不一致もしくは不統一である箇所を、修正対象事項に対応する箇所として検出するように定義されている。また、階層情報テンプレート 5 3 における修正規則は、修正対象事項修正手段 1 8 が、修正対象事項検出手段 1 7 によって検出された箇所における端子記述を、複数の階層の全てにおいて一致または統一した記述に修正するように定義されている。

## 【 0 0 3 8 】

なお、上述した階層情報テンプレート 5 3 を用いた具体的な修正動作については、図 5 ～図 8 を参照しながら後述する。また、階層情報テンプレート 5 3 は、例えば、図 5 ～図 8 のそれぞれに示すような修正規則 5 3 a ～ 5 3 d を有している。これらの修正規則 5 3 a ～ 5 3 d の詳細については後述する。

## 【 0 0 3 9 】

接続情報テンプレート 5 4 の修正対象事項は、修正対象事項検出手段 1 7 が、信号代入記述の左辺と右辺との関係が誤っている箇所を、修正対象事項に対応する箇所として検出するように定義されている。また、接続情報テンプレート 5 4 の修正規則は、修正対象事項修正手段 1 8 が、修正対象事項検出手段 1 7 によって検出された箇所における信号代入記述の左辺と右辺との関係を修正するように定義されている。なお、上述した接続情報テンプレート 5 4 を用いた具体的な修正動作については後述する。

## 【 0 0 4 0 】

非合成記述テンプレート 5 5 の修正対象事項は、修正対象事項検出手段 1 7 が、論理合成ツールが合成することのできない箇所（非合成記述箇所）を、修正対象事項に対応する箇所として検出するように定義されている。また、非合成記述テンプレート 5 5 の修正規則は、修正対象事項修正手段 1 8 が修正対象事項検出手段 1 7 によって検出された箇所を削除するように、もしくは、修正対象事項修正手段 1 8 が、修正対象事項検出手段 1 7 によって検出された箇所に、当該箇所における記述を論理合成ツールに無視させるためのディレクティブを追加・記入するように、定義される。設計者は、非合成記述箇所を削除する修正規則とディ

レクティブを追加・記入する修正規則とのいずれか一方を選択して採用することができるようにしている。

【 0 0 4 1 】

なお、上述した非合成記述テンプレート 5 5 を用いた具体的な修正動作については、図 9 を参照しながら後述する。また、非合成記述テンプレート 5 5 は、図 9 に示すような修正規則 5 5 a を有している。この修正規則 5 5 a の詳細については後述するが、図 9 に示す修正規則 5 5 a は、ディレクティブを追加・記入するように定義されたものである。

【 0 0 4 2 】

上述した H D L 字句解析手段 1 1，H D L 構文解析手段 1 2，構文的文法違反検出手段 1 3，構文的文法違反修正手段 1 4，意味論的文法違反検出手段 1 5，意味論的文法違反修正手段 1 6，修正対象事項検出手段 1 7，修正対象事項修正手段 1 8，H D L 逆構文解析手段 1 9 およびコメント付与手段 2 0 は、専用ソフトウェア（H D L 自動修正プログラム）によって実現される。

【 0 0 4 3 】

この H D L 自動修正プログラムは、例えばフレキシブルディスク，C D - R O M 等のコンピュータ読取可能な記録媒体に記録された形態で提供される。本実施形態の H D L 自動修正装置 1 は、C P U，R O M，R A M などから構成されたコンピュータ（図示省略）によって実現される。そして、R O M 等に H D L 自動修正プログラムを予め格納しておき、この H D L 自動修正プログラムを、C P U によって読み出し実行することにより、上述した各手段 1 1 ～ 2 0 としての機能が実現される。

【 0 0 4 4 】

なお、H D L 自動修正プログラムは、例えば磁気ディスク，光ディスク，光磁気ディスク等の記憶装置（記録媒体）に記録しておき、その記憶装置から通信経路を介してコンピュータに提供されてもよい。

また、テンプレート 3 0，4 0，5 1 ～ 5 5 についての情報は、設計者の手によりキーボード，マウス等を通じて入力してもよいし、別途、記録媒体を通じて入力してもよいし、上述した H D L 自動修正プログラムに予め含めた形で提供さ

れてもよい。

さらに、上述したデータベース 21～24 としては、上記 RAM を用いてもよいし、例えばフレキシブルディスク、CD-R、CD-RW 等の記録媒体を用いてもよい。

#### 【0045】

##### 〔2〕本発明の一実施形態の動作の説明

次に、上述のごとく構成された本実施形態の HDL 自動修正装置 1 の動作について、より具体的に説明する。

まず、本実施形態の HDL 自動修正装置 1 による一連の自動修正動作について簡単に説明する。

#### 【0046】

修正対象の HDL 記述（初期 HDL 記述）2A は、HDL 自動修正装置 1 に入力されると、まず、HDL 字句解析手段 11 によりトークンに分解され、トークンデータベース 21 に書き込まれる。このトークンに基づいて、HDL 構文解析手段 12 により、HDL 記述 2A の構文解析が行なわれ、HDL 記述 2A が解析木形式の記述に変換される。得られた解析木と、文法解析テンプレート 30 とに基づいて、構文的文法違反検出手段 13 により、HDL 記述 2A における構文的文法違反箇所が検出される。構文的文法違反箇所が検出された場合、その構文的文法違反箇所は、構文的文法違反修正手段 14 により、文法解析テンプレート 30 で定義された規則に従って、正しい記述に修正される。その修正結果は、HDL データベース 22 に書き込まれる。

#### 【0047】

ついで、意味論的文法違反検出手段 15 により、上述のごとく構文的文法違反箇所を修正された HDL 記述 2A の意味論解析が、HDL データベース 22 のデータに基づいて行なわれ、HDL 記述 2A における代入文の右辺と左辺の変数の型が不一致になっている部分が意味論的文法違反箇所として検出される。意味論的文法違反箇所が検出された場合、意味論的文法違反修正手段 16 により、その意味論的文法違反箇所における代入文の右辺に対して、型変換テンプレート 40 に定義された型変換関数が適用される。これにより、意味論的文法違反箇所が正

しい記述に修正され、その修正結果がHDLデータベース23に書き込まれる。

【0048】

さらに、修正対象事項検出手段17により、上述のごとく意味論的文法違反箇所を修正されたHDL記述2Aの中において、テンプレート51～55のそれぞれで定義された修正対象事項に対応する箇所が検出される。そのような箇所が検出された場合、修正対象事項修正手段18により、その箇所が、テンプレート51～55のそれぞれに定義された修正規則に従って修正され、その修正結果がHDLデータベース24に書き込まれる。

【0049】

以上のようにして各種修正を施されデータベース24に保持された、解析木形式のHDL記述は、HDL逆構文解析手段19により、通常の記述形式に変換され、修正後のHDL記述2Bとして出力される。そして、そのHDL記述2Bの修正箇所には、コメント付与手段20により、その修正に関するコメントが付与される。

【0050】

さて、次に、図2～図9を参照しながら、VHDLやVerilog-HDLによって記述されたHDL記述を修正対象として、本実施形態のHDL自動修正装置1が実際に行なう自動修正動作について説明する。ここでは、本発明の特徴的な部分である、意味論的文法違反や修正対象事項と、それらに対する修正動作と、修正結果に付与されるコメントとについて、具体的に説明する。

【0051】

〔2-1〕意味論的文法違反の修正動作について

HDL記述2Aには、VHDLのように代入文の右辺と左辺の型が不一致の場合に文法違反となるものがある。本実施形態では、そのような意味論的文法違反箇所を、意味論的文法違反検出手段15によって検出すると、型変換テンプレート40を用いて意味論的文法違反修正手段16により自動的に修正した上で、その修正内容が、コメント付与手段20により、ソース（修正後HDL記述2B）の修正箇所にコメントとして追加・記入される。

【0052】

例えば図 2 に示す初期 HDL 記述 (VHDL による記述) 2 A においては、入力変数 “a” (右辺) の型 “std\_ulogic” と、出力変数 “b” (左辺) の型 “bit” とは異なるものであるため、初期 HDL 記述 2 A 中の “b<= a ;” の箇所が、意味論的文法違反箇所として検出される。上述のような型の不一致箇所は、意味論的文法違反検出手段 1 5 が HDL データベース 2 2 を検索することによって検出される。

#### 【 0 0 5 3 】

そして、意味論的文法違反修正手段 1 6 は、その箇所を修正するために必要な型変換パターン (型変換規則, 型変換関数) が型変換テンプレート 4 0 に存在するか否かをチェックする。ここで、型変換テンプレート 4 0 においては、左辺の変数の型が “bit” で右辺の変数の型が “std\_ulogic” である場合の型変換パターン (型変換規則) “To\_bit” が、ライブラリ “std\_logic\_1164” によって提供・定義されている。

#### 【 0 0 5 4 】

従って、意味論的文法違反箇所 “b<= a ;” は、意味論的文法違反修正手段 1 6 により、修正後 HDL 記述 2 B において、“b<=To\_bit(a);” に自動的に変換され修正される。さらに、その修正箇所 “b<=To\_bit(a);” の後には、コメント付与手段 2 0 により、修正内容を示すコメント “--型変換” が付与される。

なお、型変換テンプレート 4 0 には、典型的な型変換パターン (型変換規則, 型変換関数) が予め定義されている。その型変換パターンの代表的なものを下記表 1 に示す。また、必要に応じ、型変換関数が格納されているライブラリとパッケージとを参照するための library 文と use 文の追加を自動的行なってもよい。

#### 【 0 0 5 5 】



【表 1】

左辺	右辺	ライブラリ	パッケージ
bit	std_ulogic	std_logic_1164	To_bit
bit_vector	std_logic_vector	std_logic_1164	To_bitvector
	std_ulogic_vector	std_logic_1164	To_bitvector
std_ulogic	bit	std_logic_1164	To_StdULogic
std_ulogic_vector	std_logic_vector	std_logic_1164	To_StdULogicVector
	bit_vector	std_logic_1164	To_StdULogicVector
std_logic_vector	bit_vector	std_logic_1164	To_StdLogicVector
	std_ulogic_vector	std_logic_1164	To_StdLogicVector
	integer	std_logic_arith	CONV_STD_LOGIC_VECTOR
	unsigned	std_logic_arith	CONV_STD_LOGIC_VECTOR
	signed	std_logic_arith	CONV_STD_LOGIC_VECTOR
boolean	std_ulogic	std_logic_1164	Is_X
	std_logic_vector	std_logic_1164	is_X
	std_ulogic_vector	std_logic_1164	is_X
natural	unsigned	numeric_std	TO_INTEGER
integer	signed	numeric_std	TO_INTEGER
	std_logic_vector	std_logic_signed	CONV_INTEGER
	unsigned	std_logic_arith	CONV_INTEGER
	signed	std_logic_arith	CONV_INTEGER
	std_ulogic	std_logic_arith	CONV_INTEGER
unsigned	natural	numeric_std	TO_UNSIGNED
	integer	std_logic_arith	CONV_UNSIGNED
	signed	std_logic_arith	CONV_UNSIGNED
	std_ulogic	std_logic_arith	CONV_UNSIGNED
signed	natural	numeric_std	TO_SIGNED
	integer	std_logic_arith	CONV_SIGNED
	unsigned	std_logic_arith	CONV_SIGNED
	std_ulogic	std_logic_arith	CONV_SIGNED

【0 0 5 6】

〔2-2〕言語変換規則テンプレートを用いた修正動作について

HDLを用いて回路設計を行なった場合、システム間の関係や設計フローの都合などにより、当初用いられていたHDLから異なる種類の他のHDLにHDL記述を変換することがしばしばある。このとき、当然、HDL記述が現在のHDLの言語規約を満たしていても他のHDLの言語規約を満たさない場合がある。本実施形態のHDL自動修正装置1では、上述のような変換の可能性を考慮して（他言語に変換した場合に起こりうる文法エラーを想定して）、他のHDLに変換した時にも回路設計上の問題（他言語に変換した場合の文法エラー）が生じないように、予め他のHDLの言語規約も満たすようにHDL記述を自動修正している。

【0 0 5 7】

つまり、実際に他言語へ変換するツールは存在しているが、本実施形態では、他言語への変換処理は行なわれませんが、他言語へ変換されることを考慮した他言語の言語規約チェックが行なわれ、そのチェック結果に応じた自動修正が行なわれる。このように他言語での規約チェックを事前に行なうことで、設計者に負担をかけることなく、複数種類のHDLにおいて回路設計上の問題を生じさせることのないHDL記述2Bを得ることができる。従って、いつでも他言語への変換が可能となり、また実際に他言語に変換した場合にエラー修正の手間を省くことができる。

## 【0058】

そこで、本実施形態のHDL自動修正装置1における修正対象事項検出手段17では、図3に示すように、言語変換規則テンプレート51に含まれる予約語テンプレート51a、名前テンプレート51bおよび大文字小文字ルール51dを読み込み、これらのテンプレート51a、51bや大文字小文字ルール51dに違反している箇所をチェックする。そして、違反箇所における記述は、名前生成ルール51cに従って自動的に生成された新たな記述（文字列）に変換される。また、他言語へ変換した場合の言語規約違反だけではなく、混乱を招きやすい記述に対しても、チェックおよび自動修正を行なう。

## 【0059】

例えばVerilog-HDLで記述されたHDL記述2Aに対しては、構文的文法違反検出手段13によってVerilog-HDLの予約語チェックを行なうほか、修正対象事項検出手段17によって、予約語テンプレート51aおよび名前テンプレート51bを用いて、他のHDL、例えばVHDLの予約語チェックが行なわれる。このとき、予約語テンプレート51aには、変換可能性のある言語（HDL）の予約語が全て定義・登録されており、名前テンプレート51bには、変換可能性のある言語（HDL）ごとに使用可能な端子名やネット名が登録・定義されている。

## 【0060】

図3に具体的に示すVerilog-HDLによるHDL記述2Aにおいて、“in”や“out”は、Verilog-HDLでは端子名やネット名として使用可能であるが、

VHDLでは予約語となっているため、VHDLでは、“in”や“out”を端子名やネット名として使用することは禁止されている。このように他言語に変換した場合にエラーとなる記述“in”や“out”が、修正対象事項検出手段17により修正対象の箇所として検出されると、修正対象事項修正手段18は、サフィックス、プレフィックス、連結子、追い番を記述した名前生成ルール51cを読み込み、そのルール51cに従って、新たな文字列（名前）として“in\_1”や“out\_1”をそれぞれ自動生成する。

#### 【0061】

ここで生成された新たな文字列“in\_1”や“out\_1”は、いずれも、HDL記述2Aに存在する文字列と重複しないユニークな文字列であり、且つ、Verilog-HDLやVHDLのいずれにおいても端子名やネット名として使用可能なものである。

そして、“in”や“out”は、修正後HDL記述2Bにおいて、それぞれ“in\_1”や“out\_1”に自動的に置き換えられ、さらに、その修正箇所の行末には、コメント付与手段20により、修正コメント“//修正”が付与される。

#### 【0062】

また、例えばVerilog-HDLによるHDL記述2Aにおいて、ネット名中の文字として“a \_ b”（図3参照）や“\$”を使用することは、Verilog-HDLの言語規約には違反していないが、VHDLの言語規約に違反している。そのため、HDL記述2AをVerilog-HDLからVHDLに変換した場合にエラーが発生するものと予測されるので、修正対象事項修正手段18は、名前生成ルール51aに従って、上述と同様のユニークな文字列（名前）を自動生成し、他言語に変換した場合、違反するであろうと判断した箇所を自動発生名に変更する。

図3において、HDL記述2Aの文字列“a \_ b”は、修正後HDL記述2Bでは、新たな文字列“a\_b”に自動的に置き換えられ、さらに、その修正箇所の行末に、コメント付与手段20により、修正コメント“//修正”が付与される。

#### 【0063】

さらに、Verilog-HDLでは、アルファベットの大文字と小文字を区別して使用するが、VHDLでは、アルファベットの大文字と小文字を区別しないで使

用する。このため、Verilog-HDLによるHDL記述2Aにおいて、大文字と小文字とが混在している場合、そのHDL記述2AをVerilog-HDLからVHDLに変換した場合、混乱を招いてしまう。図3に示すHDL記述2Aにおいて“a”と“A”とは区別して使用されているが、VHDLでは、これらの文字“a”および“A”を区別することができず混乱を招くことになる。

## 【0064】

そこで、本実施形態のHDL自動修正装置1では、Verilog-HDLをVHDLに変換する可能性を考慮し、修正対象事項検出手段17によって、大文字と小文字を区別して記述された同一綴りの一対の文字列“A”と“a”のうちの一方（ここでは“A”）が、修正対象事項に対応する箇所として検出され、さらに、修正対象事項修正手段18が、名前生成ルール51cに従って、上述と同様のユニークな文字列（名前）として例えば“A\_1”を自動生成する。ここで生成された新たな文字列は、HDL記述中に存在する文字列の綴りと重複しない綴りを有するものである。

## 【0065】

そして、HDL記述2Aの文字“A”は、修正後HDL記述2Bにおいて、新たな文字列“A\_1”に自動的に置き換えられ、さらに、その修正箇所の行末に、コメント付与手段20により、修正コメント“//修正”が付与される。このように“A”を“A\_1”に置き換えることにより、Verilog-HDLにおいてもVHDLにおいても、その名前を区別することが可能になる。

## 【0066】

逆に、アルファベットの大文字と小文字を区別しないVHDLで記述されたHDL記述をチェックする場合には、大文字と小文字を区別して使用するVerilog-HDLに変換する可能性を考慮し、修正対象事項検出手段17によって、HDL記述の文字列における大文字および小文字のうちの一方が、修正対象事項に対応する箇所として検出され、さらに、修正対象事項修正手段18が、検出された大文字または小文字をそれぞれ小文字または大文字に自動的に変換する。即ち、VHDLで記述されたHDL記述中の全ての文字を、小文字もしくは大文字に統一する修正が行なわれる。

## 【 0 0 6 7 】

VHDLでは、例えば端子名“b”と“B”とを同一の端子名として取り扱うが、Verilog-HDLでは、これらの端子名“b”と“B”とを異なる端子名として取り扱うため、混乱を招くことになるが、本実施形態のHDL自動修正装置1では、上述のごとく、VHDLによるHDL記述2Aにおける文字が全て小文字もしくは大文字に統一され、上述の端子名も全て“b”もしくは“B”に置き換えられる。従って、上述のような混乱を招くこともなくなる。

## 【 0 0 6 8 】

〔 2 - 3 〕 使用禁止文字情報テンプレートを用いた修正動作について

HDLを用いて回路設計を行なう際には、複数人数で設計を行なったり、ツールを使用して設計を行なったりする場合がある。このような場合、HDL記述中の名前ルールが統一されていないことがしばしばある。そのため、従来、ツール間でインターフェイスする場合や他言語に変換する場合を考慮し、手作業により名前の記述の統一をはかっていた。

## 【 0 0 6 9 】

これに対し、本実施形態のHDL自動修正装置1の修正対象事項検出手段17では、図4に示すように、使用禁止文字情報テンプレート52に含まれる使用禁止文字テンプレート52aを読み込み、使用禁止文字テンプレート52aを用いて、HDL記述2Aの使用禁止文字を使用している箇所をチェックする。そして、モジュール名、外部端子名、インスタンス名、ネット名、タイプ名、コンポーネント名、外部端子別名等において使用禁止文字が使用されていた場合、その使用禁止文字を含む文字列（名前）は、名前生成ルール52bに従って自動的に生成された新たな名前（文字列）に変換される。

## 【 0 0 7 0 】

図4では、使用禁止文字として例えば“\$”，“&”を登録された使用禁止文字テンプレート52aを用いることで、修正対象事項検出手段17により、入力Verilog-HDLで記述されたHDL記述2A中で“\$”，“&”が使用されている箇所があるかどうかチェックされる。このとき、図4に示す例では、インスタンス名として“\$”が修正対象箇所として検出され、修正対象事項修正手段

1 8 が、名前生成ルール 5 2 b（ここでは“prefix:X;”）に従って、HDL 記述 2 A に存在する文字列と重複せず且つ所定の使用禁止文字を含まない新しい文字列（名前）として、例えば“X1”を自動生成する。

#### 【 0 0 7 1 】

そして、HDL 記述 2 A の文字“\$”は、修正後 HDL 記述 2 B において、新たな文字列“X1”に自動的に置き換えられ、さらに、その修正箇所の行末に、コメント付与手段 2 0 により、修正コメント“//修正”が付与される。

これにより、例えば複数の設計者により HDL 記述 2 A を作成した場合に命名規則に違反する箇所が多数出現したとしても、設計者に負担をかけることなく、その違反箇所の名前（文字列）を容易かつ確実に修正して命名規則に従った HDL 記述 2 B を得ることができる。

#### 【 0 0 7 2 】

##### 〔 2 - 4 〕 階層情報テンプレートを用いた修正動作について

HDL 記述 2 A が複数の階層から構成されている場合（例えば図 5 参照）、各階層の端子定義記述とインスタンス部における端子記述とが不一致（不統一）であることが多々ある。このような端子記述の不一致は、設計者が文法違反の記述を行なって生じる場合や、設計者が文法的には違反していないが回路として不適切な記述をしてしまっている場合があり、これらの場合は、いずれもその不一致を解消することが望ましい。

#### 【 0 0 7 3 】

また、Verilog-HDL は、端子記述等の省略が許容される言語であるため、Verilog-HDL による HDL 記述中には、上述のような端子記述の不一致が多々ある。このような場合、上述のような不一致は回路表現として間違いではないが、安全な運用のためにできるだけ不一致の無い明快な記述を行なった方が好ましい。

#### 【 0 0 7 4 】

そこで、本実施形態の HDL 自動修正装置 1 では、

(1) 全てのモジュールポート名（端子名）をインスタンス部に記述する（図 5 参照）。

(2)上位階層におけるビット幅の記述を、下位階層におけるビット幅の記述に合わせる（図 6 参照）。

(3)上位階層におけるコンポーネントポートの記述を下位階層の記述に合わせる（図 7 参照）。

(4)インスタンス部におけるポート名（端子名）を、名前対応もしくは位置対応で統一的に記述する（図 8 参照）。

というルールを、それぞれ階層情報テンプレート 5 3 における修正規則 5 3 a ～ 5 3 d として定義する。

#### 【 0 0 7 5 】

このような修正規則 5 3 a ～ 5 3 d に従って、修正対象事項検出手段 1 7 により、HDL 記述を成す複数の階層において端子記述（ポート名記述）が不一致もしくは不統一である箇所が、修正対象事項に対応する箇所として検出される。そして、修正対象事項修正手段 1 8 により、検出された箇所における端子記述が、言語文法および修正規則 5 3 a ～ 5 3 d を定義する制御情報（階層情報テンプレート 5 3）に基づき、複数の階層の全てにおいて一致または統一した記述に自動的に修正される。

#### 【 0 0 7 6 】

階層構造をもつ Verilog-HDL による HDL 記述 2 A では、インスタンス部に、未使用の下位階層端子名を記述する必要はない。しかし、Verilog-HDL から VHDL への変換を行なう場合や、端子が未使用であることを明示したい場合には、下位階層の全ての端子名を記述した方が都合が良いので、全ての端子名（ポート名）をインスタンス部に記述するという修正規則 5 3 a (verilog instance port:adjust to module port;) を、階層情報テンプレート 5 3 に制御情報として定義・登録する。

#### 【 0 0 7 7 】

図 5 に示す HDL 記述 2 A では、下位階層において端子名として s, t, u, v の 4 つが記述されているが、上位階層では端子名 v をもつ端子が使用されていないので、インスタンス部においてその端子名 v の記述が省略されている。このような HDL 記述 2 A を HDL 自動修正装置 1 に入力すると、HDL 記述 2 A のインス

タンス部における省略記述 “`ins(.s(a),.t(b),.u(c))`” が、修正後HDL記述 2 Bにおいて、全端子名を有する記述 “`ins(.s(a),.t(b),.u(c),.v())`” に自動的に修正され、さらに、その修正箇所の行末に、コメント付与手段 2 0 により、修正コメント “//修正” が付与される。これにより、HDL記述 2 A中のインスタンス部の記述が全端子名を有する記述に変更されるので、未使用の端子を明示することができるほか、Verilog-HDLからVHDLへの変換にも対応することができるようになる。

## 【 0 0 7 8 】

図 6 に示す Verilog-HDL による HDL 記述 2 A では、下位階層において出力 `u` は 2 ビット [0:1] であることが記述されているが、上位階層では出力 `u` に対応する出力 `c` が 1 ビットであり、インスタンス部において出力 `u` のうちの未使用の 1 ビットの記述が省略されている。

## 【 0 0 7 9 】

このとき、ビット幅が不一致であった場合に上位階層のインスタンス部を下位モジュールのビット幅に合わせるという修正規則 5 3 b (`bundle port:adjust to lower module`) を、階層情報テンプレート 5 3 に制御情報として定義・登録しておけば、図 6 に示す HDL 記述 2 A は、インスタンス部における上位階層のビット幅の記述を下位階層のビット幅の記述に一致させた HDL 記述 2 B に、自動的に修正され、さらに、その修正箇所の行末に、コメント付与手段 2 0 により修正コメント “//修正” が付与される。これにより、上位階層において、出力 `u` の 2 ビットのうち 1 ビットが空いていることが明確化される。

## 【 0 0 8 0 】

なお、図 6 に示す修正後 HDL 記述 2 B では、2 ビットの信号 `d` が新たに定義され [`wire [0:1] d;`]、信号 `d` の 2 ビットのうちの一方が信号 `c` に割り当てられるとともに [`assign c = d [0] ;`]、信号 `d` と出力 `u` とが対応付けられている [`ins(.s(a),.t(b),.u(d))`]。

## 【 0 0 8 1 】

階層構造をもつ VHDL による HDL 記述 2 A においては、上位階層のコンポーネントポートの記述と下位階層のエンティティポートの記述とが不一致である



ため、上下階層を一括して処理することが不可能になる場合がある。例えば図 7 に示す H D L 記述 2 A では、下位階層においてエンティティポート名 V が記述されているが、上位階層においてそのポート名 V をもつポートが空き状態であるため、そのポート名 V の記述が省略されている。

#### 【 0 0 8 2 】

このとき、上位階層のコンポーネントポートの記述を下位階層におけるエンティティポートの記述に合わせるという修正規則 5 3 c (component port: complement to entity) を、階層情報テンプレート 5 3 に制御情報として定義・登録しておけば、図 7 に示す H D L 記述 2 A における省略記述 “port(S,T:in std\_logic;U:out std\_logic” や “port map(S=>A,T=>B,U=>C)” が、修正後 H D L 記述 2 B において、それぞれ、下位階層のエンティティポートの記述に一致させた記述 “port(S,T:in std\_logic;U,V:out std\_logic” や “port map(S=>A,T=>B,U=>C,V=>OPEN)” に自動的に修正され、さらに、その修正箇所の行末に、コメント付与手段 2 0 により修正コメント “//修正” が付与される。これにより、上位階層において、ポート名 V をもつポートの存在とそのポートが空き状態であることが明確化される。

#### 【 0 0 8 3 】

図 8 に示す H D L 記述 2 A において、Verilog-H D L (もしくは V H D L) のポート位置対応記述やポート名対応記述はいずれも文法的に適しているが、その記述を統一する必要がある場合などには、インスタンス部におけるポート名 (端子名) を名前対応もしくは位置対応で統一的に記述するという修正規則 5 3 d (port connection:name;) を、階層情報テンプレート 5 3 に制御情報として定義・登録する。

#### 【 0 0 8 4 】

例えば、初期 H D L 記述 2 A が図 8 に示すごとくポート位置対応記述されており、修正規則 5 3 d がポート名対応記述への修正を指示していた場合、H D L 記述 2 A における記述 “test1 test1\_ins(p,q);” は、“test1 test1\_ins(.a(p),.b(q));” に自動的に修正され、さらに、その修正箇所の行末に、コメント付与手段 2 0 により修正コメント “//修正” が付与される。これにより、インスタンス

部におけるポート名（端子名）が名前対応もしくは位置対応で統一的に記述される。

#### 【 0 0 8 5 】

以上のように、階層情報テンプレート 5 3 における修正対象事項や修正規則を適当に定義することにより、HDL 記述 2 A を成す複数の階層において端子記述が不一致もしくは不統一である箇所が、一致または統一した記述になるように自動修正される。従って、従来、フロントエンド（言語処理系）ではなくバックエンド（論理合成ツールや検証ツール）でないと検出できなかった不適切な記述箇所が早期に発見されて自動修正されるので、設計工程での手戻りの発生を確実に抑止することができる。

#### 【 0 0 8 6 】

##### 〔 2 - 5 〕 接続情報テンプレートを用いた修正動作について

HDL における信号代入記述は、RTL (Register Transfer Level) 記述では動作仕様の基本であるので、左辺と右辺とを間違える可能性は低い。しかし、構造記述部分では、単純であるが大量の記述が行なわれる場合があるので、左辺と右辺とを間違えて記述することも有りうる。一般に、HDL ではポート（端子）と信号代入とに関する文法が存在しており、特に、VHDL では、5 種類のポート in, out, inout, buffer, linkage に対して以下のように使用法が規定されているので、規則を、信号代入記述の左辺と右辺とへの記述可能性として表記することが可能である。

#### 【 0 0 8 7 】

in	かならず右辺でなければならない
out	かならず左辺でなければならない
inout	左辺, 右辺のどちらでもよい
buffer	左辺, 右辺のどちらでもよい
linkage	左辺, 右辺のどちらでもよい

#### 【 0 0 8 8 】

同様に、Verilog-HDL では、3 種類のポート input, output, inout に対して以下のように使用法が規定されている。

input	かならず右辺でなければならない
output	かならず左辺でなければならない
inout	左辺, 右辺のどちらでもよい

## 【 0 0 8 9 】

本実施形態のHDL自動修正装置1では、上述のような使用規則を、接続情報テンプレート54に含まれる修正規則として適用することで、修正対象事項検出手段17により、信号代入記述の左辺と右辺との関係が誤っている箇所が、修正対象事項に対応する箇所として検出され、さらに、修正対象事項修正手段18により、検出された箇所における信号代入記述の左辺と右辺との関係が、上記修正規則（使用規則）に従って自動的に修正される。

## 【 0 0 9 0 】

例えば、VHDLによる初期HDL記述2Aに、以下のような信号代入記述があった場合、文法違反ではないが、出力端子の信号bが入力端子aに代入されるのは不適切である。

```
module test(a,b);
  input a;
  output b;
  assign a=b;
endmodule
```

## 【 0 0 9 1 】

このとき、以下のような修正規則を接続情報テンプレート54に定義・登録する。

input : right	# input端子は右辺でなければならない
output: left	# output端子は左辺でなければならない
inout: both	# inout端子は両辺どちらでもよい

## 【 0 0 9 2 】

これにより、HDL記述2A内に存在する信号代入記述の各々について左辺と右辺との妥当性を確認し、逆の記述があった場合、HDL記述2Aを自動的に修正し、修正済みのHDL記述2Bを生成・出力することができる。なお、その修

正箇所の行末には、コメント付与手段20により修正コメント“//修正”が付与される。

【0093】

修正後のHDL記述2Bは以下のようになる。

```
module test(a,b);
  input a;
  output b;
  assign b=a;//修正
endmodule
```

【0094】

以上のように接続情報テンプレート54における修正対象事項や修正規則を適当に定義することにより、信号代入記述の左辺と右辺との関係が誤っている箇所を正しい関係に自動修正することができる。また、HDLの言語種類によって修正規則を使い分けることで、それぞれの言語仕様に応じた修正を自動的行なうことが可能になる。

【0095】

〔2-6〕非合成記述テンプレートを用いた修正動作について

論理合成したいHDL記述においては、論理検証時に使用した論理合成不能な波形観測用シミュレーション記述等がコメントアウトされずに残っている場合が多々ある。

【0096】

そこで、本実施形態のHDL自動修正装置1では、非合成記述テンプレート55に含まれる修正規則55a（図9参照）において、論理合成不能な波形観測用シミュレーション記述文を全て列挙するとともに、その記述文を削除するか、もしくは、その記述文を論理合成ツールに無視させるためのディレクティブを追加・記入するかを指定する。なお、ディレクティブは、該当記述文の前後を囲むように記述され、このようにディレクティブで囲まれた記述文は、論理合成ツールによって無視され論理合成対象外の記述として取り扱われる。なお、該当記述文を削除するかディレクティブで囲むかは設計者によって適宜選択される。

【0097】

例えば図9では、非合成記述テンプレート55に含まれる修正規則55aにおいて、論理合成不能な波形観測用シミュレーション記述文として“initial文”と“\$monitor( )”とが列挙され、これらの記述文を“synthesis on/off directive”で囲むように指定されている。

【0098】

図9に具体的に示すHDL記述2Aにおいて、“initial o = 1' b0;”と“\$monitor(o);”とが論理合成不能な波形観測用シミュレーション記述文であり、これらの記述文が修正対象事項検出手段17により検出される。そして、修正対象事項修正手段18により、その記述文の前後に“synthesis on/off directive”が自動的に追加・記入され（図9の修正後HDL記述2B参照）、さらに、その追加・記入箇所（修正箇所）の行末に、コメント付与手段20により修正コメント“//修正”が付与される。

【0099】

これにより、HDL記述2Aにおいて、例えば、論理検証時に使用した論理合成不能な波形観測用シミュレーション記述文“initial o = 1' b0;”や“\$monitor(o);”がコメントアウトされずに残っていたとしても、このような記述文が論理合成ツールにおいて問題（エラー）を引き起こすことがなくなる。従って、上述のような記述文を設計者が手作業により削除するといった作業が不要になり、設計者にかかる負担を大幅に削減することができる。

【0100】

### 〔3〕本発明の一実施形態の効果の説明

このように、本発明の一実施形態としてのHDL自動修正装置1によれば、初期HDL記述2Aに存在する不適切な記述を検査することが可能となり、さらに修正を施されたHDL記述2Bが生成されるので、高品質のHDL記述2Bを得ることができる。

【0101】

特に、本実施形態では、重度の意味論的文法違反が自動修正され且つその修正箇所が明確化されるので、設計者にかかる負担が大幅に軽減されるとともに、高

品質のHDL記述2Bを得ることができる。また、文法違反ではないが回路設計上考慮すべき箇所（修正対象事項に対応する箇所）が適切な記述に自動修正され且つその修正箇所が明確化されるので、設計者にかかる負担が大幅に軽減されるとともに、より高品質のHDL記述2Bを得ることができる。

#### 【0102】

さらに、テンプレート51～55における修正対象事項や修正規則を適当に定義することにより、回路設計上考慮すべき箇所（不適切な記述）や設計者の単純ミスを早期に発見して確実に自動修正することができるので、設計工程での手戻りの発生を確実に抑止することが可能になる。

不適切な記述は多くの場合、設計者の設計意図とは異なる回路の機能や構造を表現しているので、後続の論理検証や回路実装の工程になって誤りが発見され修正作業が発生するが、本実施形態のHDL自動修正装置1を用いて、不適切な記述をHDL記述の作成時に修正することにより、無駄な修正作業を発生させずに済む。

#### 【0103】

一方、修正に関するコメント（注意書き）を修正箇所に付与することにより、修正箇所が明確化され、設計者は、HDL記述2Bのどこをどのように修正したかを容易に視認することができる。従って、設計者は、自動修正結果が意図するものかどうか、即ち適正な修正が行なわれたか否かの確認作業を容易かつ確実に行なえ、設計者にかかる負担が大幅に削減される。

#### 【0104】

また、修正後HDL記述2Bに付与された修正コメントにより、HDLによって回路設計情報を記述する際に、どのような箇所において、修正が必要な記述をする可能性があるかを設計者に把握させることができるので、設計者に対する教育的効果も得られる。

#### 【0105】

##### 〔4〕その他

なお、本発明は上述した実施形態に限定されるものではなく、本発明の趣旨を逸脱しない範囲で種々変形して実施することができる。

例えば、上述した実施形態では、HDLがVHDLやVerilog-HDLである場合について説明したが、本発明は、これらのVerilog-HDLやVHDL以外の言語にも適用可能であり、その場合も上述と同様の作用効果が得られ、回路設計分野やソフトウェア開発分野における効率化・負担削減に大きく寄与するものである。

【0106】

〔5〕 付記

（付記1） ハードウェア記述言語（以下、HDLという）によって記述された回路設計情報（以下、HDL記述という）を自動的に修正する装置であって、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段と、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段と、

該HDL構文解析手段による解析結果に基づいて該HDL記述の意味論解析を行ない、該HDL記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段と、

前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を、型変換規則として定義する型変換テンプレートと、

該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代入文の右辺に対して、該型変換テンプレートに定義された該型変換関数を適用することにより、該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段と、

該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段と、

該意味論的文法違反修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段とをそなえて構成されたことを特徴とする、HDL自動修正装置。

【0107】

(付記 2) 文法違反ではないが回路設計上考慮すべき修正対象事項、および、該修正対象事項に対応する箇所の修正規則を定義する制御情報テンプレートと、

該 H D L 構文解析手段による解析結果に基づき、該 H D L 記述において、該修正対象事項に対応する箇所を検出する修正対象事項検出手段と、

該修正対象事項検出手段によって検出された箇所を、該制御情報テンプレートに定義された該修正規則に従って修正する修正対象事項修正手段とをさらにそなえ、

該 H D L 逆構文解析手段が、該意味論的文法違反修正手段および該修正対象事項修正手段による修正を施された該 H D L 記述の逆構文解析を行なうとともに、

該コメント付与手段が、該意味論的文法違反修正手段および該修正対象事項修正手段によって修正を施された箇所に、該コメントを付与することを特徴とする、付記 1 記載の H D L 自動修正装置。

【 0 1 0 8 】

(付記 3) ハードウェア記述言語 (以下、H D L という) によって記述された回路設計情報 (以下、H D L 記述という) を自動的に修正する装置であって、

修正対象の H D L 記述の字句解析を行なう H D L 字句解析手段と、

該 H D L 字句解析手段による解析結果に基づいて該 H D L 記述の構文解析を行ない、該 H D L 記述を解析木形式の記述に変換する H D L 構文解析手段と、

文法違反ではないが回路設計上考慮すべき修正対象事項、および、該修正対象事項に対応する箇所の修正規則を定義する制御情報テンプレートと、

該 H D L 構文解析手段による解析結果に基づき、該 H D L 記述において、該修正対象事項に対応する箇所を検出する修正対象事項検出手段と、

該修正対象事項検出手段によって検出された箇所を、該制御情報テンプレートに定義された該修正規則に従って修正する修正対象事項修正手段と、

該修正対象事項修正手段による修正を施された該 H D L 記述の逆構文解析を行ない、該 H D L 記述を前記解析木形式の記述から通常の記述形式に変換する H D L 逆構文解析手段と、



該修正対象事項修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段とをそなえて構成されたことを特徴とする、HDL自動修正装置。

## 【0109】

(付記4) 該修正対象事項検出手段が、処理中の該HDLを異なる種類の他のHDLに変換した場合に該他のHDLの言語規約を満たさない箇所を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該修正対象事項検出手段によって検出された箇所を、該他のHDLの言語規約を満たす記述に変換・修正するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記2または付記3に記載のHDL自動修正装置。

## 【0110】

(付記5) 処理中の該HDLがアルファベットの大文字と小文字を区別して使用するものである場合、処理中の該HDLを、大文字と小文字を区別しないで使用する他のHDLに変換する可能性を考慮し、

該修正対象事項検出手段が、大文字と小文字を区別して記述された同一綴りの一对の文字列のうち的一方を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該HDL記述中に存在する文字列の綴りと重複しない綴りの新しい文字列を生成してから、該修正対象事項検出手段によって検出された該一方の文字列を該新しい文字列に置換するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記4に記載のHDL自動修正装置。

## 【0111】

(付記6) 処理中の該HDLがアルファベットの大文字と小文字を区別しないで使用するものである場合、処理中の該HDLを、大文字と小文字を区別して使用する他のHDLに変換する可能性を考慮し、

該修正対象事項検出手段が、該HDL記述の文字列における大文字および小文

字のうちの一方を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該修正対象事項検出手段によって検出された大文字または小文字をそれぞれ小文字または大文字に変換するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記 4 記載の H D L 自動修正装置。

#### 【 0 1 1 2 】

(付記 7) 該修正対象事項検出手段が、所定の使用禁止文字を含む文字列を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該 H D L 記述に存在する文字列と重複せず且つ該所定の使用禁止文字を含まない新しい文字列を生成してから、該修正対象事項検出手段によって検出された該使用禁止文字を含む文字列を該新しい文字列に置換するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記 2 ～付記 6 のいずれか一つに記載の H D L 自動修正装置。

#### 【 0 1 1 3 】

(付記 8) 該修正対象事項検出手段が、該 H D L 記述を成す複数の階層において端子記述が不一致もしくは不統一である箇所を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該修正対象事項検出手段によって検出された箇所における該端子記述を、該複数の階層の全てにおいて一致または統一した記述に修正するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記 2 ～付記 7 のいずれか一つに記載の H D L 自動修正装置。

#### 【 0 1 1 4 】

(付記 9) 該修正対象事項検出手段が、信号代入記述の左辺と右辺との関係が誤っている箇所を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該修正対象事項検出手段によって検出された箇所

における該信号代入記述の左辺と右辺との関係を修正するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記 2 ～付記 8 のいずれか一つに記載の H D L 自動修正装置。

【 0 1 1 5 】

(付記 1 0) 該修正対象事項検出手段が、論理合成ツールが合成することのできない箇所を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該修正対象事項検出手段によって検出された箇所を削除するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記 2 ～付記 9 のいずれか一つに記載の H D L 自動修正装置。

【 0 1 1 6 】

(付記 1 1) 該修正対象事項検出手段が、論理合成ツールが合成することのできない箇所を、該修正対象事項に対応する箇所として検出するように、該制御情報テンプレートにおける該修正対象事項が定義されるとともに、

該修正対象事項修正手段が、該修正対象事項検出手段によって検出された箇所に、当該箇所における記述を該論理合成ツールに無視させるためのディレクティブを追加・記入するように、該制御情報テンプレートにおける該修正規則が定義されることを特徴とする、付記 2 ～付記 9 のいずれか一つに記載の H D L 自動修正装置。

【 0 1 1 7 】

(付記 1 2) ハードウェア記述言語（以下、H D L という）によって記述された回路設計情報（以下、H D L 記述という）を、コンピュータに自動修正させるための H D L 自動修正プログラムであって、

修正対象の H D L 記述の字句解析を行なう H D L 字句解析手段、

該 H D L 字句解析手段による解析結果に基づいて該 H D L 記述の構文解析を行ない、該 H D L 記述を解析木形式の記述に変換する H D L 構文解析手段、

該 H D L 構文解析手段による解析結果に基づいて該 H D L 記述の意味論解析を行ない、該 H D L 記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段、

該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代入文の右辺に対して、前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を適用することにより、該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段、

該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段、および、

該意味論的文法違反修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段として、該コンピュータを機能させることを特徴とする、HDL自動修正プログラム。

【0118】

(付記13) 該HDL構文解析手段による解析結果に基づき、該HDL記述において、文法違反ではないが回路設計上考慮すべき修正対象事項に対応する箇所を検出する修正対象事項検出手段、および、

該修正対象事項検出手段によって検出された箇所を、予め定義された該修正対象事項に対応する箇所の修正規則に従って修正する修正対象事項修正手段として、該コンピュータをさらに機能させ、

該HDL逆構文解析手段に、該意味論的文法違反修正手段および該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行なわせるとともに、

該コメント付与手段に、該意味論的文法違反修正手段および該修正対象事項修正手段によって修正を施された箇所に、該コメントを付与させることを特徴とする、付記12記載のHDL自動修正プログラム。

【0119】

(付記14) ハードウェア記述言語(以下、HDLという)によって記述された回路設計情報(以下、HDL記述という)を、コンピュータに自動修正させるためのHDL自動修正プログラムであって、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行

ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段、

該HDL構文解析手段による解析結果に基づき、該HDL記述において、文法違反ではないが回路設計上考慮すべき修正対象事項に対応する箇所を検出する修正対象事項検出手段、

該修正対象事項検出手段によって検出された箇所を、予め定義された該修正対象事項に対応する箇所の修正規則に従って修正する修正対象事項修正手段、

該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段、および、

該修正対象事項修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段として、該コンピュータを機能させることを特徴とする、HDL自動修正プログラム。

【0120】

(付記15) ハードウェア記述言語(以下、HDLという)によって記述された回路設計情報(以下、HDL記述という)を、コンピュータに自動修正させるためのHDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体であって、

該HDL自動修正プログラムが、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段、

該HDL構文解析手段による解析結果に基づいて該HDL記述の意味論解析を行ない、該HDL記述における代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段、

を、型変換規則として定義する型変換テンプレート、

該意味論的文法違反検出手段によって該意味論的文法違反箇所とみなされた代入文の右辺に対して、前記代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を適用することにより、該意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段、

該意味論的文法違反修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段、および、

該意味論的文法違反修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段として、該コンピュータを機能させることを特徴とする、HDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体。

【0121】

(付記16) 該HDL自動修正プログラムが、

該HDL構文解析手段による解析結果に基づき、該HDL記述において、文法違反ではないが回路設計上考慮すべき修正対象事項に対応する箇所を検出する修正対象事項検出手段、および、

該修正対象事項検出手段によって検出された箇所を、予め定義された該修正対象事項に対応する箇所の修正規則に従って修正する修正対象事項修正手段として、該コンピュータをさらに機能させ、

該HDL逆構文解析手段に、該意味論的文法違反修正手段および該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行なわせるとともに、

該コメント付与手段に、該意味論的文法違反修正手段および該修正対象事項修正手段によって修正を施された箇所に、該コメントを付与させることを特徴とする、付記15記載のHDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体。

【0122】

(付記17) ハードウェア記述言語(以下、HDLという)によって記述された回路設計情報(以下、HDL記述という)を、コンピュータに自動修正させるためのHDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体であって、

該HDL自動修正プログラムが、

修正対象のHDL記述の字句解析を行なうHDL字句解析手段、

該HDL字句解析手段による解析結果に基づいて該HDL記述の構文解析を行ない、該HDL記述を解析木形式の記述に変換するHDL構文解析手段、

該HDL構文解析手段による解析結果に基づき、該HDL記述において、文法違反ではないが回路設計上考慮すべき修正対象事項に対応する箇所を検出する修正対象事項検出手段、

該修正対象事項検出手段によって検出された箇所を、予め定義された該修正対象事項に対応する箇所の修正規則に従って修正する修正対象事項修正手段、

該修正対象事項修正手段による修正を施された該HDL記述の逆構文解析を行ない、該HDL記述を前記解析木形式の記述から通常の記述形式に変換するHDL逆構文解析手段、および、

該修正対象事項修正手段によって修正を施された箇所に、該修正に関するコメントを付与するコメント付与手段として、該コンピュータを機能させることを特徴とする、HDL自動修正プログラムを記録したコンピュータ読取可能な記録媒体。

【0123】

【発明の効果】

以上詳述したように、本発明のHDL自動修正装置（請求項1～3）およびHDL自動修正プログラム（請求項4）並びに同プログラムを記録したコンピュータ読取可能な記録媒体（請求項5）によれば、以下のような効果ないし利点を得ることができる。

（1）重度の意味論的文法違反が自動修正され且つその修正箇所が明確化されるので、設計者にかかる負担が大幅に軽減されるとともに、高品質のHDL記述を得ることができる（請求項1，4，5）。

【0124】

（2）文法違反ではないが回路設計上考慮すべき箇所が適切な記述に自動修正され且つその修正箇所が明確化されるので、設計者にかかる負担が大幅に軽減されるとともに、高品質のHDL記述を得ることができる（請求項2，3）。

（3）制御情報テンプレートにおける修正対象事項や修正規則を適当に定義することにより、回路設計上考慮すべき箇所（不適切な記述）や設計者の単純ミス

を早期に発見して確実に自動修正することができるので、設計工程での手戻りの発生を確実に抑止することが可能になる（請求項 2，3）。

## 【0125】

（4）修正に関するコメントを修正箇所が付与することにより、設計者は、HDL 記述のどこをどのように修正したかを容易に視認することができる。従って、設計者は、自動修正結果が意図するものかどうか、即ち適正な修正が行なわれたか否かの確認作業を容易かつ確実に行なえ、設計者にかかる負担が大幅に削減される（請求項 1～5）。

## 【0126】

（5）修正に関するコメントを修正箇所が付与することにより、HDL によって回路設計情報を記述する際に、どのような箇所において、修正が必要な記述をする可能性があるかを設計者に把握させることができ、設計者に対する教育的効果も得られる（請求項 1～5）。

## 【0127】

（6）制御情報テンプレートにおける修正対象事項や修正規則を適当に定義することにより、HDL 記述を処理中の HDL から他の HDL に変換する可能性がある場合、他の HDL に変換した時に回路設計上の問題が生じないように、予め他の HDL の言語規約も満たすように HDL 記述を自動修正することができる。従って、設計者に負担をかけることなく、複数種類の HDL において回路設計上の問題を生じさせることのない HDL 記述を得ることができる。

## 【0128】

（7）制御情報テンプレートにおける修正対象事項や修正規則を適当に定義することにより、使用禁止文字を含む文字列が、HDL 記述に存在する文字列と重複せず且つ使用禁止文字を含まない新しい文字列に自動修正される。従って、例えば複数の設計者により HDL 記述を作成した場合に命名規則に違反する箇所が多数出現したとしても、設計者に負担をかけることなく、その違反箇所の名前（文字列）を容易かつ確実に修正して命名規則に従った HDL 記述を得ることができる。

## 【0129】



(8) 制御情報テンプレートにおける修正対象事項や修正規則を適当に定義することにより、HDL記述を成す複数の階層において端子記述が不一致もしくは不統一である箇所が、一致または統一した記述になるように自動修正される。従って、従来、フロントエンド（言語処理系）ではなくバックエンド（論理合成ツールや検証ツール）でないと検出できなかった不適切な記述箇所が早期に発見されて自動修正されるので、設計工程での手戻りの発生を確実に抑止することができる。

【0130】

(9) 制御情報テンプレートにおける修正対象事項や修正規則を適当に定義することにより、信号代入記述の左辺と右辺との関係が誤っている箇所を正しい関係に自動修正することができる。

【0131】

(10) 制御情報テンプレートにおける修正対象事項や修正規則を適当に定義することにより、論理合成ツールが合成することのできない箇所を自動的に削除したり、その箇所における記述を論理合成ツールに無視させるためのディレクティブを自動的に追加・記入したりすることができる。これにより、HDL記述において、例えば、論理検証時に使用した論理合成不能な波形観測用シミュレーション記述等がコメントアウトされずに残っていたとしても、このような記述が論理合成ツールにおいて問題を引き起こすことがなくなる。従って、上述のような記述を設計者が手作業により削除するといった作業が不要になり、設計者にかかる負担を大幅に削減することができる。

【図面の簡単な説明】

【図1】

本発明の一実施形態としてのHDL自動修正装置の構成を示すブロック図である。

【図2】

本実施形態の動作を説明するための図である。

【図3】

本実施形態の動作を説明するための図である。

【図 4】

本実施形態の動作を説明するための図である。

【図 5】

本実施形態の動作を説明するための図である。

【図 6】

本実施形態の動作を説明するための図である。

【図 7】

本実施形態の動作を説明するための図である。

【図 8】

本実施形態の動作を説明するための図である。

【図 9】

本実施形態の動作を説明するための図である。

【符号の説明】

- 1   HDL 自動修正装置
- 2 A   修正対象のHDL記述（初期HDL記述）
- 2 B   修正後のHDL記述
- 1 1   HDL字句解析手段
- 1 2   HDL構文解析手段
- 1 3   構文的文法違反検出手段
- 1 4   構文的文法違反修正手段
- 1 5   意味論的文法違反検出手段
- 1 6   意味論的文法違反修正手段
- 1 7   修正対象事項検出手段
- 1 8   修正対象事項修正手段
- 1 9   HDL逆構文解析手段
- 2 0   コメント付与手段
- 2 1   トークンデータベース
- 2 2～2 4   HDLデータベース
- 3 0   文法解析テンプレート

4 0 型変換テンプレート

5 1 言語変換規則テンプレート（制御情報テンプレート）

5 1 a 予約語テンプレート

5 1 b 名前テンプレート

5 1 c 名前生成ルール

5 1 d 大文字小文字ルール

5 2 使用禁止文字情報テンプレート（制御情報テンプレート）

5 2 a 使用禁止文字テンプレート

5 2 b 名前生成ルール

5 3 階層情報テンプレート（制御情報テンプレート）

5 3 a ～ 5 3 d 修正規則

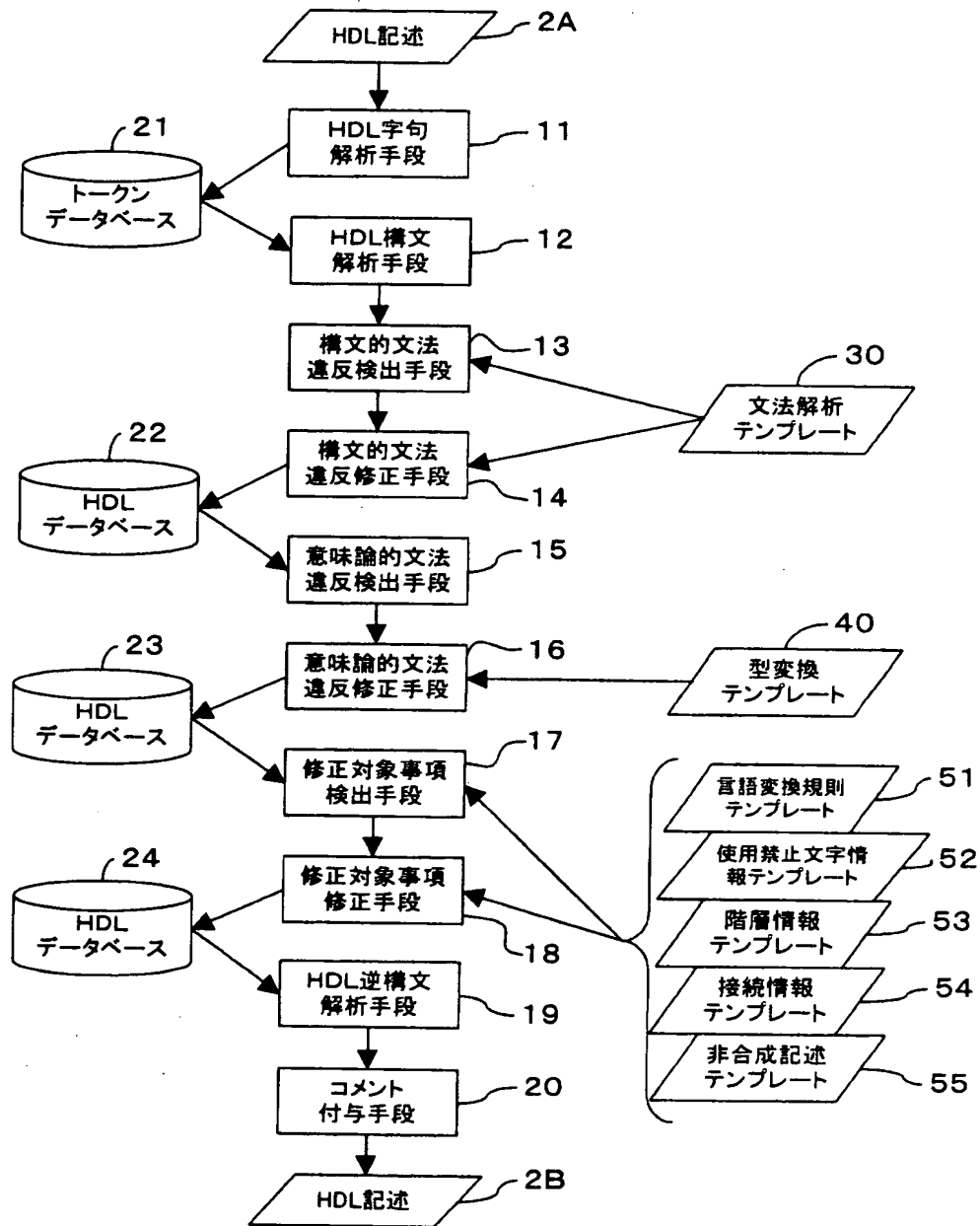
5 4 接続情報テンプレート（制御情報テンプレート）

5 5 非合成記述テンプレート（制御情報テンプレート）

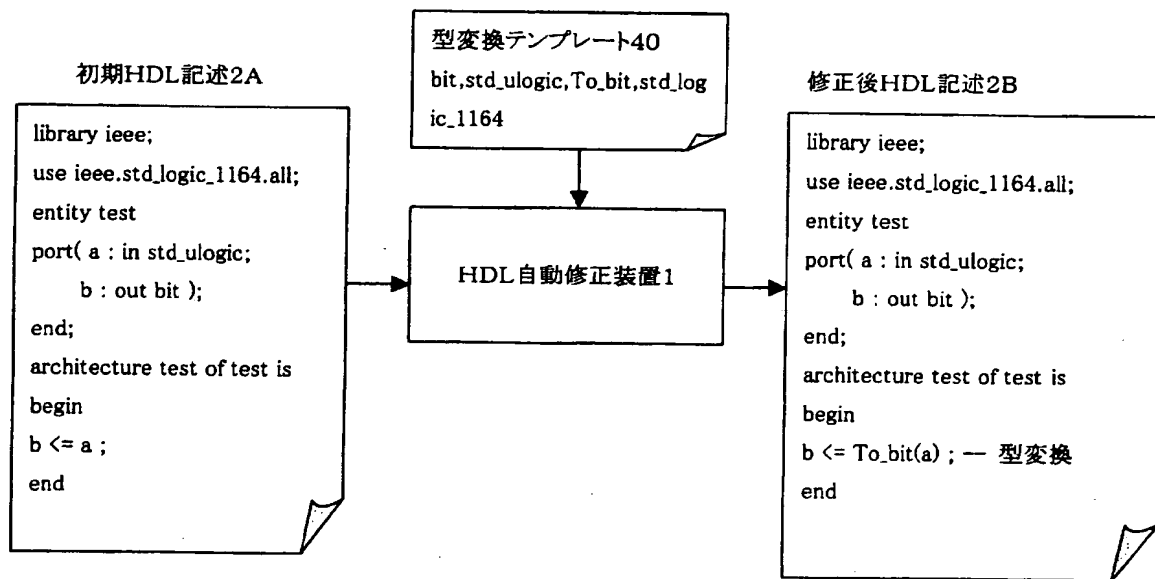
5 5 a 修正規則

【書類名】 図面

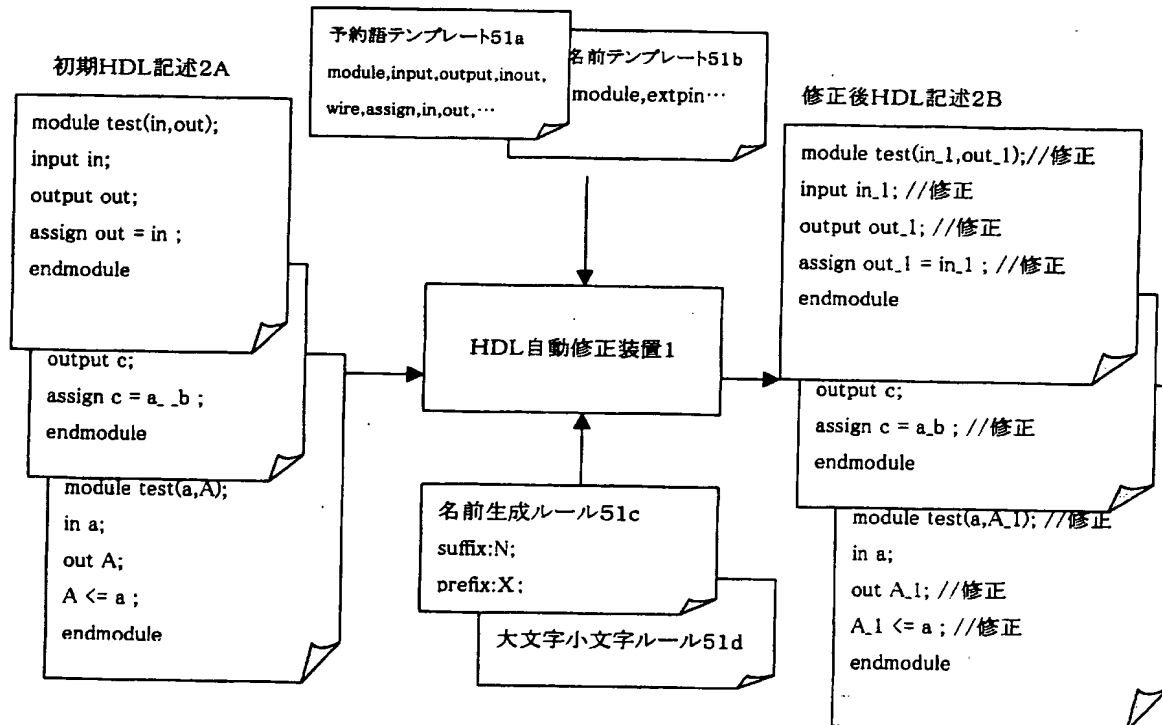
【図 1】



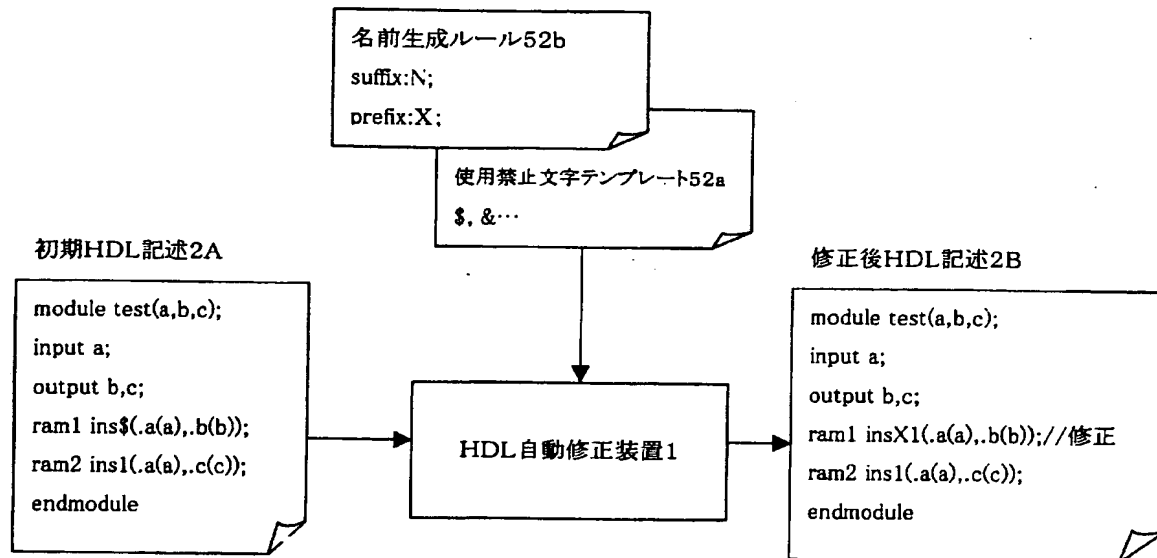
【図 2】



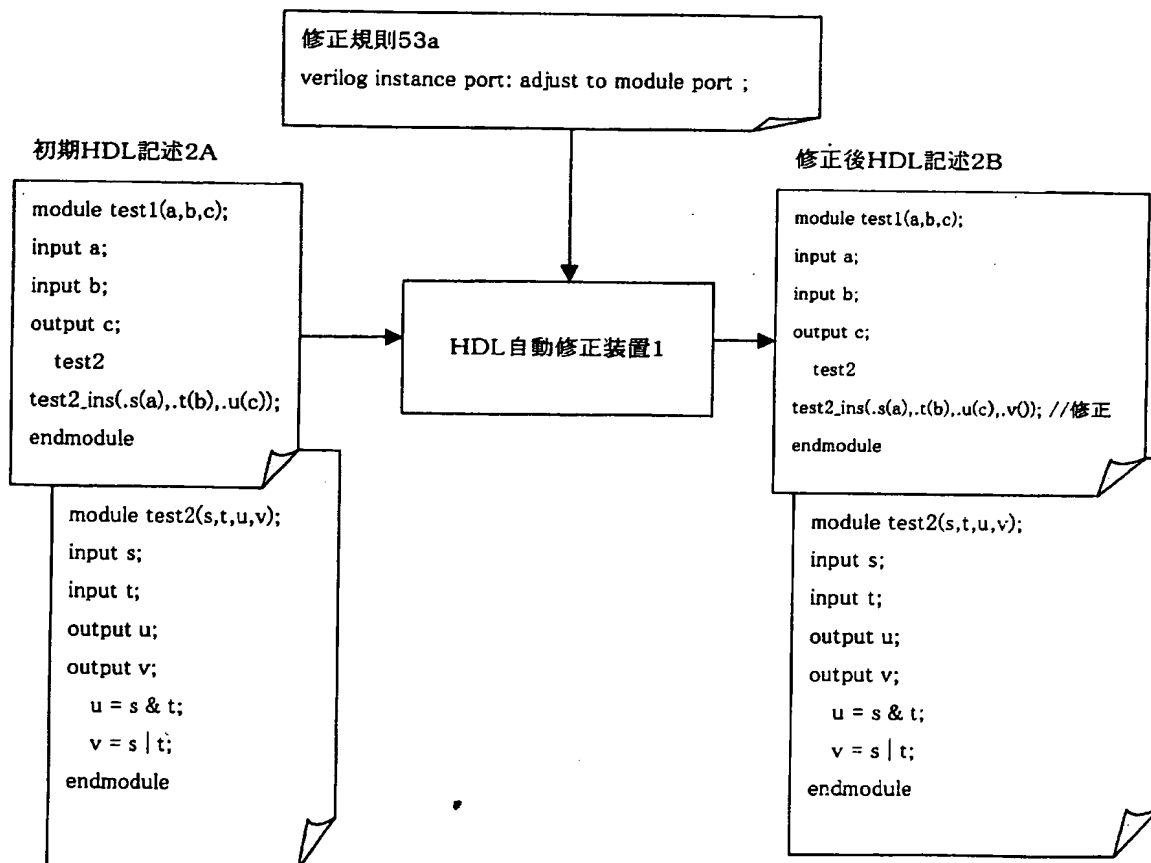
【図 3】



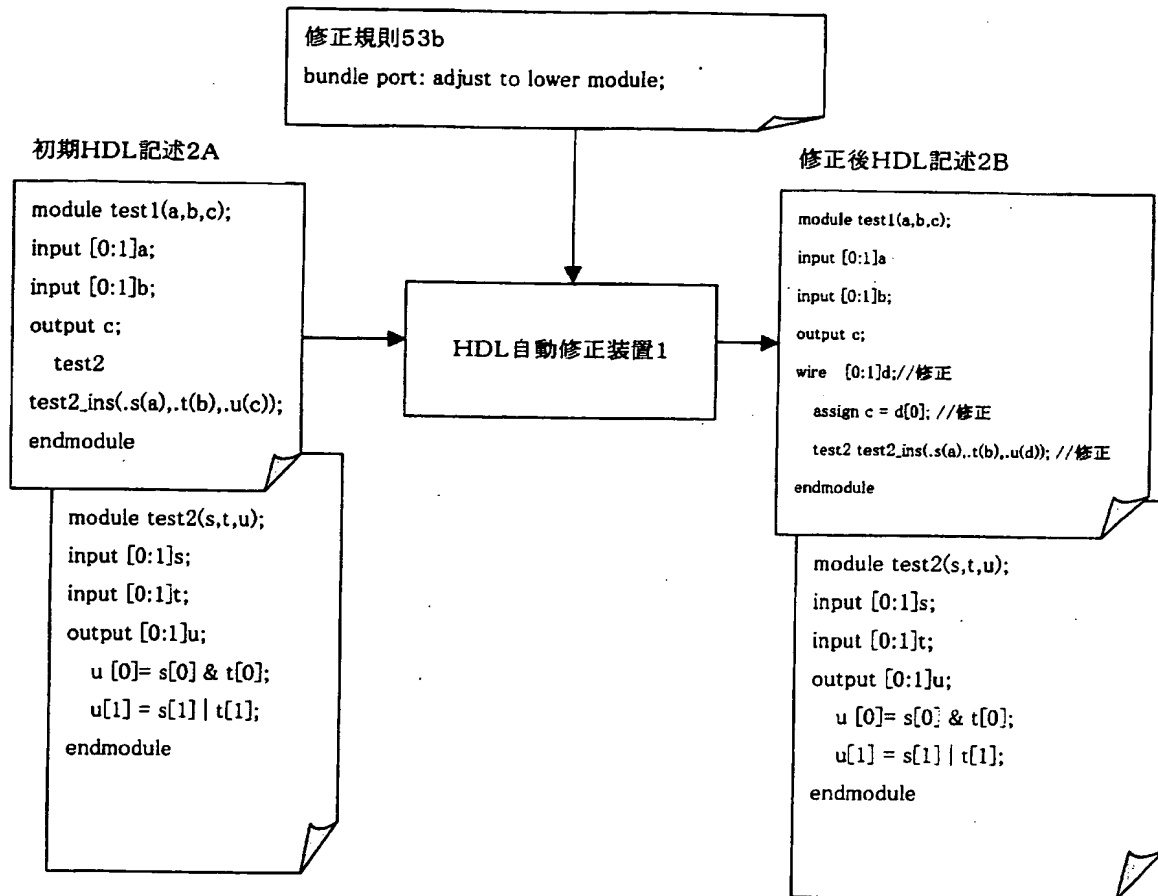
【図 4】



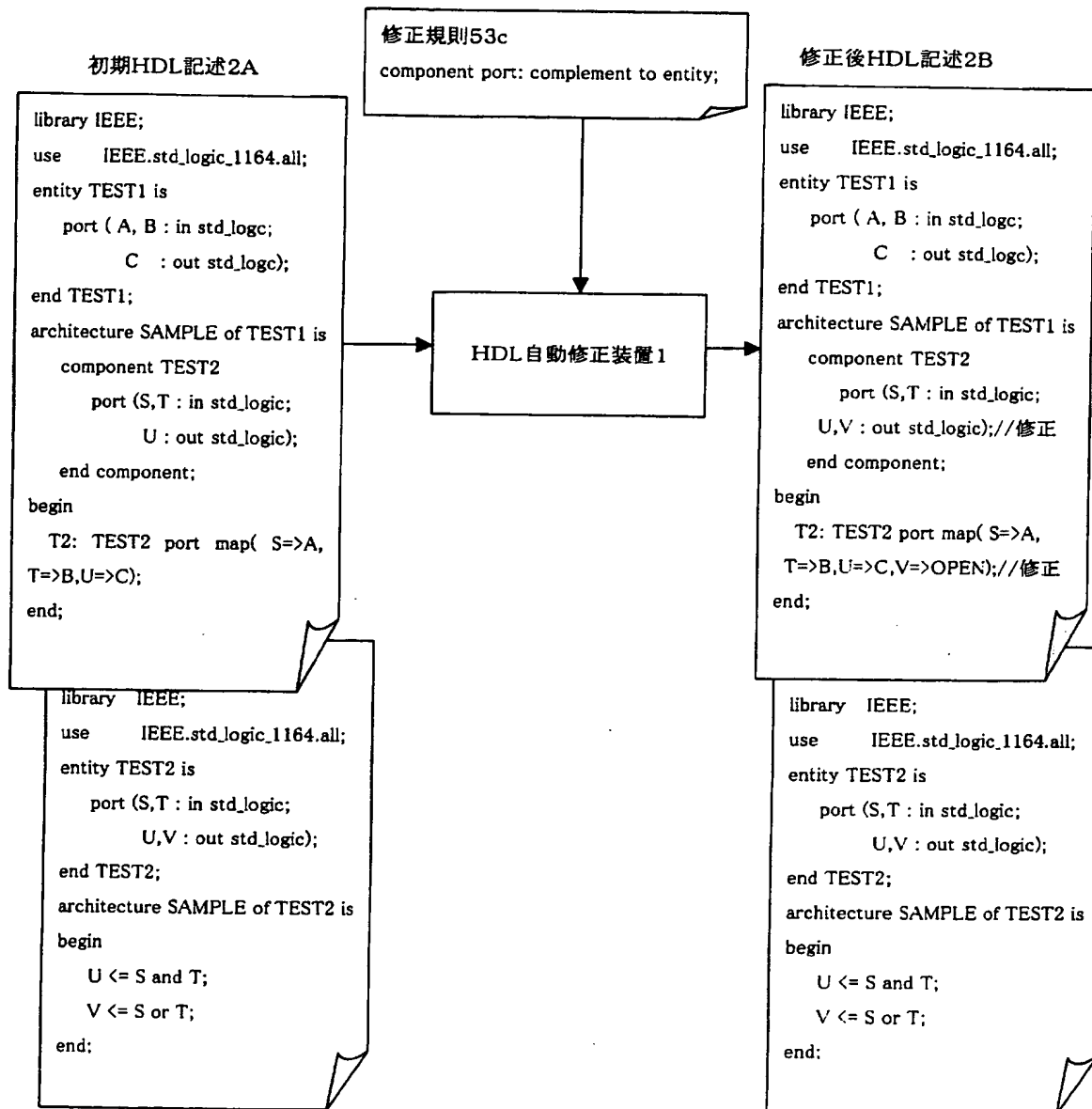
【図 5】



【図 6】

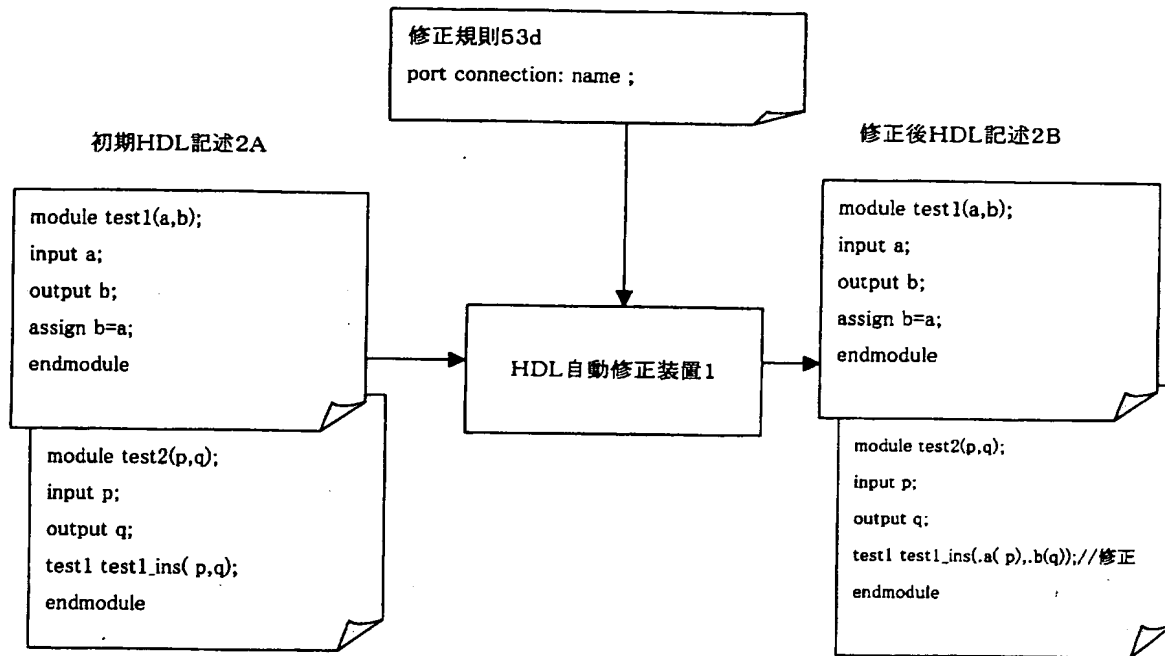


【図 7】

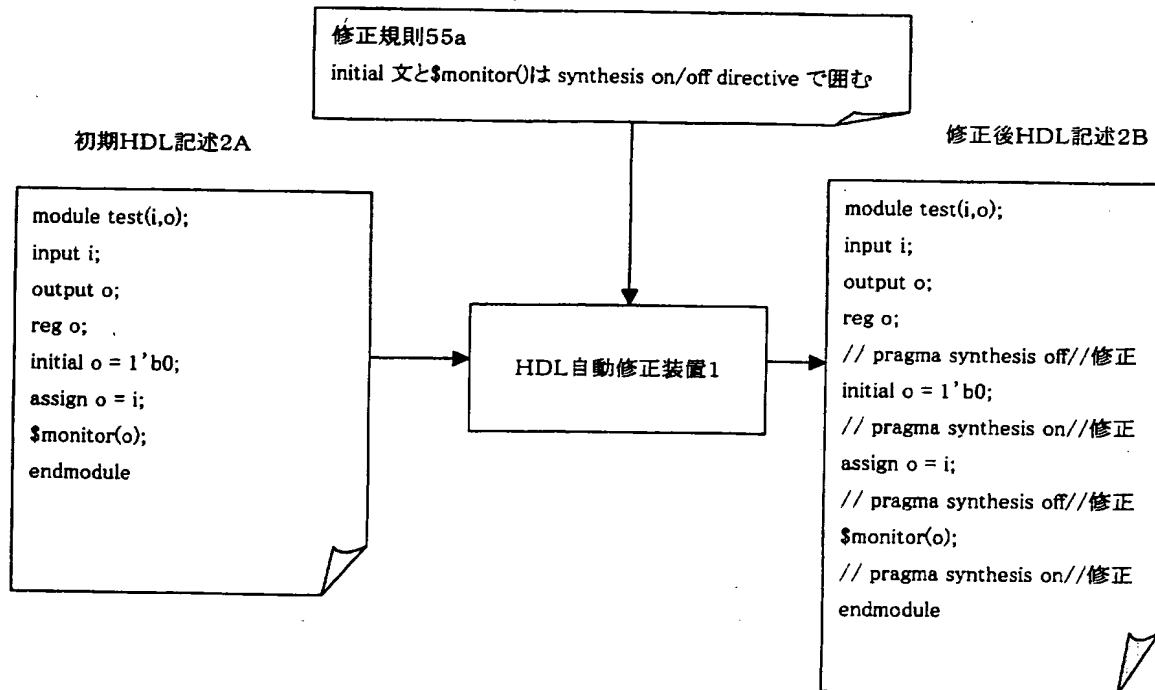




【図 8】



【図 9】



【書類名】 要約書

【要約】

【課題】 重度の意味論的文法違反を自動修正し且つその修正箇所が明確になるようにして、設計者にかかる負担を大幅に軽減するとともに、高品質のHDL記述が得られるようにする。

【解決手段】 HDL字句解析手段11やHDL構文解析手段12のほか、代入文の右辺と左辺の変数の型が不一致になっている部分を意味論的文法違反箇所として検出する意味論的文法違反検出手段15と、代入文の右辺の変数の型を左辺の変数の型に変換する型変換関数を定義する型変換テンプレート40と、意味論的文法違反箇所とみなされた代入文の右辺に対して前記型変換関数を適用して意味論的文法違反箇所を正しい記述に修正する意味論的文法違反修正手段16と、修正後のHDL記述の逆構文解析を行なう逆構文解析手段19と、修正箇所に修正に関するコメントを付与するコメント付与手段20とをそなえて構成する。

【選択図】 図1

出 願 人 履 歴 情 報

識別番号 [000005223]

1. 変更年月日 1996年 3月26日

[変更理由] 住所変更

住 所 神奈川県川崎市中原区上小田中4丁目1番1号

氏 名 富士通株式会社